Software Engineering Institute

Results of SEI Independent Research and Development Projects

Dio de Niz, Sherman Eagles, Peter H. Feiler, John Goodenough, Jörgen Hansson, Paul Jones,
Rick Kazman, Mark Klein, Prof Insup Lee, Gabriel Moreno, Robert Nord, Ipek Ozkaya, Daniel Plakosh,
Raj Rajkumar, Lui Sha, Robert Stoddard, Kurt Wallnau, Charles B. Weinstock, and Lutz Wrage

December 2008

TECHNICAL REPORT
CMU/SEI-2008-TR-025
ESC-TR-2008-025

http://www.sei.cmu.edu

Carnegie Mellon

This report was prepared for the

SEI Administrative Agent
ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2100

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

For information about purchasing paper copies of SEI reports, please visit the publications section of our website (http://www.sei.cmu.edu/publications/).

Table of Contents

List of Figures

List of Tables

List of Equations

Abstract

The Software Engineering Institute (SEI) annually undertakes several independent research and development (IRAD) projects. These projects serve to (1) support feasibility studies investigating whether further work by the SEI would be of potential benefit and (2) support further exploratory work to determine whether there is sufficient value in eventually funding the feasibility study work as an SEI initiative. Projects are chosen based on their potential to mature and/or transition software engineering practices, develop information that will help in deciding whether further work is worth funding, and set new directions for SEI work. This report describes the IRAD projects that were conducted during fiscal year 2008 (October 2007 through September 2008).

1 Introduction

1.1 Purpose of the SEI Independent Research and Development Program

Software Engineering Institute (SEI) independent research and development (IRAD) funds are used in two ways: (1) to support feasibility studies investigating whether further work by the SEI would be of potential benefit and (2) to support further exploratory work to determine whether there is sufficient value in eventually funding the feasibility study work as an SEI initiative. It is anticipated that each year there will be three or four feasibility studies and that one or two of these studies will be further funded to lay the foundation for the work possibly becoming an initiative.

Feasibility studies are evaluated against the following criteria:

- Mission criticality: To what extent is there a potentially dramatic increase in maturing and/or transitioning software engineering practices if work on the proposed topic yields positive results? What will the impact be on the Department of Defense (DoD)?
- Sufficiency of study results: To what extent will information developed by the study help in deciding whether further work is worth funding?
- New directions: To what extent does the work set new directions as contrasted with building on current work? Ideally, the SEI seeks a mix of studies that build on current work and studies that set new directions.

1.2 Overview of IRAD Projects

The following research projects were undertaken in FY 2008:

- A Software System Engineering Approach for Fault Containment
  Peter H. Feiler, Dio de Niz, Lutz Wrage, Jörgen Hansson, Lui Sha, and Raj Rajkumar
- Understanding the Relationship of Cost, Benefit, and Architecture
  Ipek Ozkaya, Rick Kazman, and Mark Klein
- Mechanism Design
  Daniel Plakosh, Mark Klein, Gabriel Moreno, and Kurt Wallnau
- Assurance Cases for Medical Devices
  John Goodenough, Charles B. Weinstock, Paul Jones, Prof Insup Lee and Sherman Eagles
- Modeling Stakeholder Requirements for Integrated Use in Both Process Improvement and Product Development
  Robert Stoddard and Robert Nord

These projects are summarized in this technical report.

2 A Software System Engineering Approach for Fault Containment

Peter H. Feiler, Dio de Niz, Lutz Wrage, Jörgen Hansson, Lui Sha, and Raj Rajkumar

2.1 Purpose

Why do system-level failures still occur despite the use of fault tolerance techniques? The lack of effective system-level fault management and stability solutions, despite best efforts at fault tolerance, is a major challenge in modern avionics and aerospace. System engineering approaches, in the form of hardware redundancy for managing hardware failures, are well-established. However, providing a *software* system engineering approach for systematic fault management with predictable results remains a challenge. The following examples illustrate the point.

After years of development, F-22 flight tests began in late 1997. But the airplane still experienced serious avionics instability problems as late as 2003:

The Air Force told us avionics have failed or shut down during numerous tests of F/A-22 aircraft due to software problems. The shutdowns have occurred when the pilot attempts to use the radar, communication, navigation, identification, and electronic warfare systems concurrently.[1]

The workload generated by different system configurations affects the execution characteristics of the system in ways that are difficult to trace. The workload may introduce instability due to violation of assumptions made by application software components about the timing and fault characteristics of the application data streams they operate on, which can lead to a failure.

The European Space Agency's Ariane 5 rocket exploded during its maiden flight. The destruction was triggered by the overflow of the horizontal velocity variable in a reused Ariane 4 software component to perform a function that was "not required for Ariane 5."[2] That is, a legacy feature that was not even needed destroyed the rocket. This is a dramatic example of system instability, that is, failure due to inconsistent system configuration: a fault in an unneeded function was not contained and cascaded into a total system failure. The reason for the overflow was the representation of a vertical velocity value as a 16-bit integer thereby placing a range restriction on the value that was exceeded by Ariane 5. It could and should have been a minor fault that would have no impact on the flight if the fault in the unneeded function had been contained there.

When laptops with a dual-core processor came out, ITunes fails crashed. ITunes was designed as multi-threaded application, but until the dual-core processor became available, only one thread at a time was executing. Two concurrently executing threads were attempting to update the same music catalog without explicit synchronization. Similarly, a well-established concurrency control protocol called Priority Ceiling Protocol (PCP) will fail on dual-core processors as it assumes that only one thread will be executing, which is not the case if a thread run queue is shared between the two processor cores.

[1] GAO Testimony to Committee on Armed Service http://www.gao.gov/new.items/d03603t.pdf

[2] Ariane 5 Flight 501 http://en.wikipedia.org/wiki/Ariane_5_Flight_501

When systems become virtualized, assumptions about physical redundancy will be violated. The DARPA-net had five physical trunk lines as its backbone; they became five logical trunk lines on one fiber-optic cable when telecom companies went fiber optic. Within three months this fiber optic cable was dug up and damaged, resulting in two DARPA-nets for one week.

Control systems perform sampled processing of signal streams from sensors to actuators. Sampling jitter induced by different scheduling policies and by the way communication between tasks is implemented affects the stability of controllers, as shown by a benchmark study by Cervin et. al.[3] Figure 2-1 illustrates the impact of different scheduling and communication timing on a highly unstable control system.

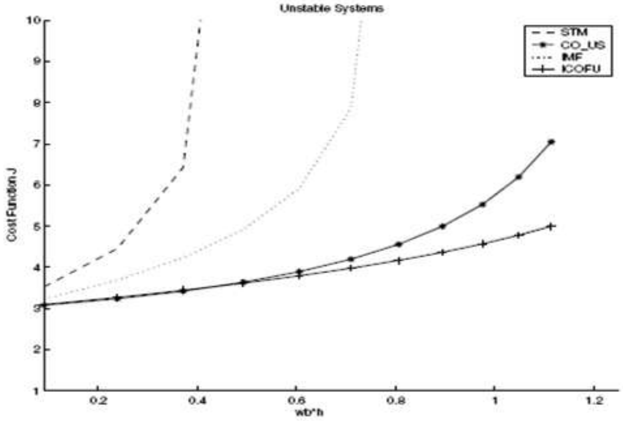


Figure 2-1: Impact of Scheduling Algorithm on Controller Stability

Figure 2-2 illustrates different dimensions of this problem space wherein different engineering roles make assumptions about the context in which they are used. As these assumptions are often undocumented and not validated during the development of a system, mismatches are often not detected until system integration, acceptance testing, or operation.

[3] "Control Loop Timing Analysis Using TrueTime and Jitterbug," A. Cervin, et. al., Proceedings IEEE CCACSD 2006.

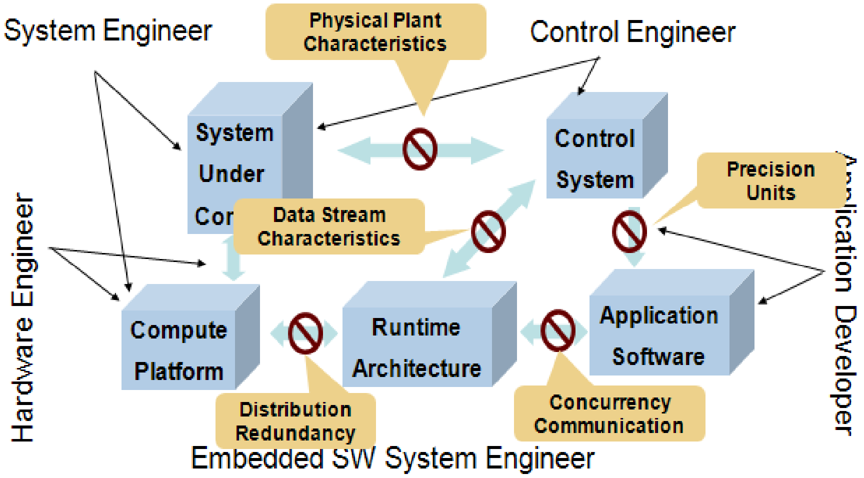


Figure 2-2 : Mismatched Assumptions

The objective of the project has been to identify system fault behaviors that are not addressed by component-fault containment techniques, to develop a formalized analysis framework for system-fault containment and stability management, and to validate system architectures in the context of this framework. The focus of this system-fault containment and stability management framework is on system-level consistency characteristics and rules for identifying direct or indirect contributions to their violation by individual components and by infrastructure services. By extending best practices, including architecture modeling and analysis, architecture patterns, component- and system-level fault tolerance, and design rules, we will provide developers what is required to place future developments on a sound theoretical footing.

2.1.1 Background

The National Coordination Office for Networking and Information Technology Research and Development (NITRD) has, in its work on high-confidence software and systems, identified research that needs to be conducted; specifically, five technology goals that must be met to realize the vision of high-confidence software systems[4] (see the report for more details on specificity of the each technology):

(i) Provide a sound theoretical, scientific, and technological basis for assured construction of safe, secure systems. (ii) Develop hardware, software, and system engineering tools that incorporate ubiquitous, application-based, domain-based, and risk-based assurance. (iii) Reduce the effort, time, and cost of assurance and quality certification processes. (iv) Provide a technology base of public domain, advanced-prototype implementations of high-confidence technologies to enable rapid adoption. (v) Provide measures of results.

Virtual machines have been recognized as a key concept for providing robustness through fault containment in integrated modular avionics systems. Known as partitioned architecture in the

[4] National Office for Networking and Information Technology Research (NITRD), High Confidence Software and Systems Coordinating Group, “High confidence software and systems research needs”, 2001 (available at www.nitrd.gov/pitac/), see pp 8—10.

avionics systems community, this mechanism provides time and space partitioning to isolate application components and subsystems from affecting each other due to resource sharing. This architecture pattern can be found in the ARINC 653 standard.[5] In a recent study of the migration of an avionics system from a federated system architecture to a partitioned system architecture, the PCS team identified sources of previously absent system-level faults due to different age characteristics of data streams under the partitioned system runtime architecture.[6]

Steve Vestal from Honeywell has demonstrated that impact analysis based on models of the runtime architecture, that is, the application system deployed on an execution platform, can be the basis for isolation analysis and fault propagation modeling. Through error model and fault occurrence annotations, he demonstrated the feasibility of reliability and fault tree analysis from the same architecture model that was the basis of global schedulability analysis.[7] His experience has led to incorporating the concept of error propagation into the error model annex of the Society of Automotive Engineers (SAE) Architecture Analysis and Design Language (AADL) standard. Similarly, the DARP initiative at York University has utilized architecture dependency information to perform fault propagation analysis.[8]

Loni Welch, in his Desiderata work, has investigated a scalable resource management approach for distributed real-time systems in the context of the DD(X) program.[9] His approach focuses on managing the desirable performance characteristics of critical information flows in a distributed embedded application system. The requirement for support of end-to-end flow specifications in support of system-level consistency analysis has been raised by the Future Combat System (FCS) system architecture contractor and other avionics and aerospace contractors.

Lui Sha, while a member of technical staff at the SEI, investigated an innovative approach to managing software fault tolerance in light of dependable system upgrade. This approach overcomes shortcomings of redundancy by replication through an analytically redundant fault container mechanism for software components that are control system applications (Simplex).[10] In a DARPA-funded collaborative project (John Lehoczky, Raj Rajkumar, Bruce Krogh [Carnegie Mellon University], Lui Sha and Peter Feiler [SEI], and Jon Preston [Lockheed Martin]) this technology was applied to an avionics system. In the context of this project, it was recognized that component-level fault containment can still lead to system-level inconsistencies that result in

[5] Avionics Application Software Standard Interface, ARINC 653 Standard Document, www.arinc.com.

[6] P. H. Feiler, D. P. Gluch, J. J. Hudak, B. A. Lewis , “Pattern-Based Analysis of an Embedded Real-time System Architecture”, IFIP TC-2 Workshop on Architecture Description Languages (WADL), World Computer Congress, Aug. 22-27, 2004, Toulouse, France, Series: IFIP International Federation for Information Processing , Vol. 176, 2005, ISBN: 0-387-24589-8]

[7] P. Binns and S. Vestal ,”Hierarchical Composition and Abstraction in Architecture Models”, IFIP TC-2 Workshop on Architecture Description Languages (WADL), World Computer Congress, Aug. 22-27, 2004, Toulouse, France, Series: IFIP International Federation for Information Processing , Vol. 176, 2005, ISBN: 0-387-24589-8.

[8] Modular Architectural Representation and Analysis of Fault Propagation and Transformation, Proceedings of FESCA, ENTCS 141(3), April 2005.

[9] L. Welch, B. Shirazi, B., and B. Ravindran, “DeSiDeRaTa: QoS Management Technology For Dynamic, Scalable, Dependable, Real-Time Systems”, in Proceedings of the 15th Symposium on Distributed Computer Control Systems (DCCS'98), IFAC, Sept. 1998.

[10] L. Sha, JB Goodenough and B. Pollack., “Simplex Architecture: Meeting the Challenges of Using COTS in High-Reliability Systems”, Crosstalk, April 1998.

faulty behavior of other components. Under the guidance of Peter Feiler, Jun Li investigated, in a Ph.D. thesis, the feasibility of capturing relevant characteristics of component interactions that would lead to system-level inconsistencies.[11]

The PERFORM group at the University of Illinois at Urbana-Champaign (UIUC), led by Prof. William H. Sanders, conducts research in the design and validation of dependable and secure networked systems. Such systems often have requirements for high performance, dependability, and security, and these goals may contradict one another. By providing a unified method to validate system performance, dependability, and security during the entire design process, the group develops and applies sound engineering principles to large-scale system design advanced modeling, analysis, and simulation environment. [12]

2.2 Approach

We have divided the project into three phases:

- root cause identification of system wide fault propagation
- development of analytical frameworks to predict the impact of seemingly minor faults on the system operation
- pilot of the analysis framework on industrial systems and develop architecture design guidance to reduce such faults

Root cause identification involved identification of high priority and high criticality system failures due to unexpected fault propagation and of contributing factors of such failures. This activity draws on Lockheed Martin's experience with several fighter aircraft developments, in particular the F-16, F-22, and F-35, input from industry through the ARTIST2 Workshop on Integrated Modular Avionics (IMA),[13] and evaluation of problem history data from the Carnegie Mellon University team of the DARPA Urban Grand Challenge (UGC).

The development of an analytical framework focused on four root cause areas that have been identified in Phase 1. This analytical framework lead to a tool-based validation method of system architecture in each of these root cause areas. We have applied these tool-based analyses on architecture models of actual, industrial-embedded, software-intensive systems.

The insights from the analytical framework allowed us to define and analyze architecture patterns that are aimed at addressing robustness and stability in systems. The patterns included redundancy patterns, partitioned systems architecture patterns, and end-to-end flow patterns.

We have chosen AADL as a basis for these analysis frameworks for model-based engineering because of its strength to be (i) non-ambiguously and objectively human readable *and* (ii)

[11] P. H. Feiler and J. Li. Managing inconsistency in reconfigurable systems. In IEE Proceedings Software, pages 172--179, 1998

[12] D. Deavours, G. Clark, T. Courtney, D. Daly, S. Derasavi, J. Doyle, W.H. Sanders and P. Webster, "The Möbius framework and its implementation" IEEE Transactions on Software Engineering, 28(10):956–970, 2002.

[13] ARTIST2 Network of Excellence on Embedded Systems Design Workshop on Integrated Modular Avionics, Nov 2007, http://www.artist-embedded.org/artist/Integrated-Modular-Avionics.html.

processable and analyzed by machines due to well-defined semantics. This system-level approach to fault containment has significant technical and programmatic merits that complement those of a component-level fault containment approach. Technically, system stability is achieved by a combination of component-level fault containment and well-formed dependency at the system level. Component-level fault containment ensures the safe sharing of hardware and logical services. That is, a component's faults cannot corrupt other components' code and data and cannot over use its CPU quota; nor can a component's faults corrupt the common OS and middleware services. An AADL-based system engineering approach enforces design rules for well-formed dependency, meaning that we can verify that a component may use, but not depend on, the service of a less critical component. Well-formed dependency is a key to system dependability by preventing a minor fault cascade into a major failure.

2.3 Collaborations

We have utilized the existing collaboration between Lockheed Martin Corporation and Prof. Lui Sha (UIUC), the SEI team's previous collaboration with Prof. Rajkumar (Carnegie Mellon University) and Prof. Rajkumar's participation in the Carnegie Mellon team of the DARPA Urban Grand Challenge, as well as Dr. Feiler's collaboration with the avionics industry through his role as the technical lead of the SAE AADL standard.

During the project, Dr. Lui Sha's doctoral students have been collaborating with the SEI team on issues related to virtual processors and conducting related research in the medical device domain. During the project, we have also collaborated with Prof. Rajkumar and his doctoral students to address issues regarding virtualization of time. Finally, Dr. Feiler has had the opportunity to spend three months at École Nationale Superieure Telecom (ENST), a Technical University in Paris, France to collaborate with several Ph.D. students and Prof. Hugues and Prof. Pautet on issues of validated runtime system generation and integration of safety and security concerns into partitioned architectures.

2.4 Evaluation Criteria

The key criteria for evaluating this project have been the ability to identify several root cause areas of system-wide fault propagation, the ability to develop or adapt existing frameworks to predictably contributors to those root causes through analysis of architecture models, and to codify guidance in architecture patterns.

In addition, our objective has been to demonstrate the practicality of the analysis framework by applying it and its supporting toolset to industrial pilot projects.

2.5 Results

As systems have become more software-intensive, there is an increasing risk in software system integration causing system-level problems.

2.5.1 Root Cause

We have identified four root-cause areas of system-wide faults that are not addressed by traditional fault tolerance techniques:

- Partitions as isolation regions: Partitioned architectures, as promoted by the ARINC653 standard, offer a virtual processor concept that provides both time and space partitioning—giving the illusion of exclusively dedicated hardware. Large-scale embedded applications can be modularized into partitions, known as integrated modular avionics (IMA) in the avionics domain, and deployed on a range of distributed-compute platforms. Experience with actual systems has shown that use of the partition concept can still lead to performance issues due to unplanned resource sharing across partitions. We characterized several actual problem scenarios in the use of partitions that have been encountered with the F-35 through the use of AADL models. This allowed us to pin-point the key contributors to the unexpected reduction of performance in both the application and in the runtime infrastructure.
- Violation of data stream assumptions: Control engineers make assumptions about the physical systems being observed and controlled. They create models of their algorithms and the controlled systems at various fidelity levels. These models are analyzed to gain confidence in the stability of the system. Application developers translate these control equations into application software components that execute in a real-time system environment in discrete time, in many cases distributed across multiple processors. These implementation choices affect the assumptions made by control engineers about the latency, latency jitter, and age of the data processed by the control loop, resulting in unexpected instability of the control behavior. We have identified a number of contributors to end-to-end latency variations due to choices in the implementation of embedded system in software that result in potential control system instability.
- Inconsistent system state due to nondeterminism: Modern avionics architectures are multi-threaded to increase utilization of individual processors through techniques, such as rate-monotonic analysis, and to take advantage of concurrency in distributed and multiple processor hardware platforms. Since much of avionics system processing is periodic in nature, it is tempting to perform all processing through periodic sampling. This leads to issues when discrete events are to be processed, such as service requests through switches, push buttons, or menu entries on multi-function displays. Similarly, different parts of the system transition between different operational modes and are reconfigured due to faults or based in user request. In many legacy systems and some new designs, such processing is performed by sampling system state. Migration to multithreaded, partitioned, and distributed architectures introduces concurrency, which if not managed properly results in unexpected non-deterministic behavior. For example, events and service requests may be ignored and hand-shaking sequences in protocols such as weapons release protocols may lock up.
- Virtualization of time and resources: Control system and mission system processing is time sensitive. During our analysis of avionics systems, we have identified virtualization of timelines when migrating to partitioned systems as a second contributor to the nondeterministic signal stream processing behavior. This virtualization of timelines, when

used by communication mechanisms, results in multiple independent time reference points (clocks). The use of globally asynchronous locally synchronous (GALS) architectures in some avionics systems, such as the F22, has the same effect. Experiences with autonomous vehicles, such as those used in the DARPA Urban Grand Challenge (UGC), have confirmed that multiple time reference points are a major system-fault root cause area.

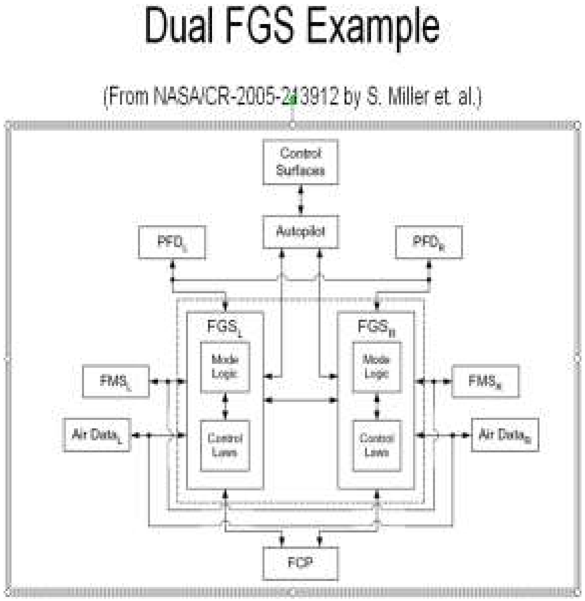
We have developed analytical frameworks to address each of these root cause areas.

- We have developed a fault impact analysis framework to model and validate fault propagation. This framework is based on and extends the Fault Propagation Calculus (FPC) developed by Wallace at University of York.[14] We have mapped FPC into AADL and the Error Model Annex standard. This work extends the original work by providing traceability between fault sources and affected system components. In addition, the original calculus does not take into account the hardware platform, the partitioned architecture concept, and architecture dynamics. We have identified an approach to address those in the AADL-based fault impact analysis framework and created an initial prototype in the AADL toolset. We have started a collaboration with WW Technology, a company that has developed an initial commercial prototype of a fault propagation analysis capability with a user-friendly and intuitive interface for engineers under Small Business Innovation Research (SBIR) funding. The benefit of such a capability has been illustrated on a customer project. The intent of this collaboration is to integrate our work with WW Technology's under SBIR Phase 2 funding to provide a scalable solution and to interface with reliability analysis.
- We have developed a flow latency analysis framework that takes into account design decisions regarding task dispatching and scheduling, the choice of communication mechanisms, and partitioned architectures, such as ARINC 653, to determine the impact on end-to-end latency and latency jitter, that is, decisions regarding the runtime architecture of the embedded software system. The UIUC team has complemented our work with an analysis capability that focuses on determining the latency contributions of the hardware network and bus architecture. The flow latency analysis method has been applied to several industrial avionics system models.
- We have developed an analytical framework that allows us to explore concurrency issues in discrete event processing nondeterminism that can be introduced due to concurrency, distributed processing, and synchronous processing. We have demonstrated the feasibility of using chaotic system theory and model checkers, such as Alloy, as a computer-based solution. We have applied chaotic system theory developed by Ortmeier et al.[15] as a way of exploring execution and communication ordering issues due to the introduction of concurrency to several examples from the avionics and automotive domains. We have

[14] Wallace, M.: Modular Architectural Representation and Analysis of Fault Propagation and Transformation. In Electronic Notes in Theoretical Computer Science 141 (2005) 53-71.

[15] F. Ortmeier, A. Thums, G. Schellhorn, and W. Reif. Combining Formal Methods and Safety Analysis: The Formosa Approach. Integration of Software Specification Techniques for Applications in Engineering. Part V: Verification. Lecture Notes in Computer Science. 2004. pp 474-493.

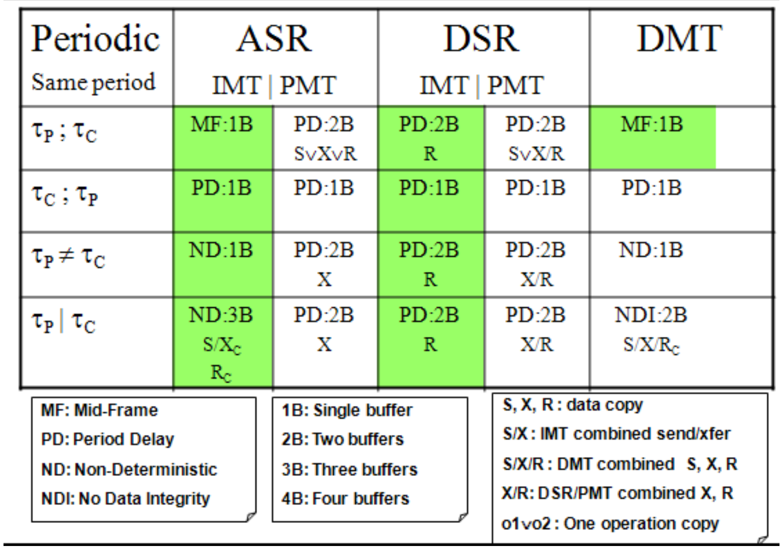
utilized the AADL annex concept to extend AADL in support of concurrency constraint specifications. Based on this experimental platform, we then developed specific analysis capabilities to identify concurrency issues under certain assumptions, such as use of lossless protocols or race conditions in event processing. In particular we have extended a model checking approach for determining the validity of the mode logic in a dual-redundant distributed system when operating as a globally asynchronous system. A dual redundant flight guidance system was originally analyzed through model checking by a Rockwell-Collins and University of Minnesota team for synchronous and asynchronous systems (see Figure 2-3). We mapped this application into an AADL model and identified issues of loss of events due to sampled monitoring of state variables to observe events due to concurrency and asynchronicity. We have proposed an initial approach to systematically evolve an architecture pattern from a synchronous solution on a single processor to a distributed globally asynchronous implementation that is property preserving.


Dual FGS Example

(From NASA/CR-2005-213912 by S. Miller et. al.)

Figure 2-3: Mode Logic Validation of a Dual Redundant System

- We have developed a lattice framework that allows us to reason about the impact of virtualization of time on time-sensitive data and event processing. In addition, the SAE AADL standard has been revised under the leadership of Dr. Feiler to include the concept of synchronization domains to support modeling of globally synchronous systems. The revised version of this standard has been approved by 29 voting members from the industrial community. Our work on virtualization of time has also led to a property-preserving method for generating highly efficient port-to-port communication (see Figure 2-4), which assures the preservation of deterministic timing semantics of sampled processing crucial to minimizing end-to-end latency jitter.



Periodic Same period	ASR IMT	 PMT	DSR IMT	 PMT	DMT
$\tau_P ; \tau_C$	MF:1B	PD:2B SvXvR	PD:2B R	PD:2B SvX/R	MF:1B
$\tau_C ; \tau_P$	PD:1B	PD:1B	PD:1B	PD:1B	PD:1B
$\tau_P \neq \tau_C$	ND:1B	PD:2B X	PD:2B R	PD:2B X/R	ND:1B
$\tau_P \mid \tau_C$	ND:3B S/X_C R_C	PD:2B X	PD:2B R	PD:2B X/R	NDI:2B $S/X/R_C$

MF: Mid-Frame
PD: Period Delay
ND: Non-Deterministic
NDI: No Data Integrity

1B: Single buffer
2B: Two buffers
3B: Three buffers
4B: Four buffers

S, X, R : data copy
S/X : IMT combined send/xfer
S/X/R : DMT combined S, X, R
X/R: DSR/PMT combined X, R
o1vo2 : One operation copy

Figure 2-4: Property Preserving Port Buffer Optimization

We have applied these analytical frameworks in the context of several industrial system models. One set of models represent several U.S. Army helicopter architectures, all in the process of migrating to a partitioned architecture, partially or fully adhering to ARINC 653. We have applied the analytical framework to a reference architecture in a NASA/JPL project. We have teamed with the Aerospace Vehicle Systems Institute (AVSI) industry consortium of avionics companies from the United States and Europe, including Boeing, Lockheed Martin, Airbus, Rockwell-Collins, BAE Systems, GE Aviation, FAA, and the U.S. Army in the System Architecture Virtual Integration (SAVI) project. AADL and its toolset has been chosen to perform a proof of concept (POC) demonstration of architecture-centric, model-based engineering through predictive analysis and continuous validation of operational quality attributes to reduce system-level faults currently discovered during system integration, acceptance testing, and operation. The analytical frameworks we have developed are part of the analysis capabilities deployed in this project.

We have also developed and validated several architecture patterns that address some of the system-level fault impact issues. These patterns include duel redundancy observer and guard patterns (with and without voting), N-Version programming, recovery block (Simplex), and validated the mode logic of an operator-managed dual-redundancy pattern. These patterns are becoming part of a handbook on architecture modeling with AADL.

2.6 Publications and Presentations

Presentations have been given in a number of forums, including the UIUC AADL workshop (Dec 2006), the SAE AADL Standards User Group meeting (Jan 2007, July 2007), the Open Group RT-Forum Workshop (Jan, April, July, Oct 2007, Jan, April, Sept 2008), the International Workshop on Aspect-Oriented Modeling (March 2007), the Army Advisory Group (Oct 2007), the ARTIST2 Network of Excellence on Embedded Systems Design Workshop on Integrated Modular Avionics (Nov 2007), the International Congress on Embedded Real-Time Systems (Jan 2008), Workshop on Cyber-Physical Systems (May and Nov 2008), AADL Starter Workshop at

PEO Aviation and AVSI (Sept, Oct 2008), US Army Embedded Real-Time Systems Workshop (Oct 2008).

The following are publications related to this project. All conference papers were accompanied by a presentation.

[de Niz 2007]
de Niz, Dionisio & Feiler, Peter H. "Aspects in the Industry Standard AADL." *Proceedings of 10th International Workshop on Aspect-Oriented Modeling*. Vancouver, Canada, 2007.

[de Niz 2008]
de Niz, Dionisio. "Architectural Concurrency Equivalence with Chaotic Models." *5th International Workshop on Model-based Methodologies for Pervasive and Embedded Software*. 2008.

[de Niz 2008b]
de Niz, Dionisio & Feiler, Peter. "On Resource Allocation in Architectural Models." *Proceedings of the 11th IEEE International Symposium on Object/service-oriented Real-time distributed Computing*. Orlando, FL, 2008.

[Feiler 2007a]
Feiler, Peter H. "Integrated Modular Avionics: The Good, The Bad, and The Ugly." *ARTIST2 Network of Excellence on Embedded Systems Design Workshop on Integrated Modular Avionics, Proceedings*. http://www.artist-embedded.org/artist/Integrated-Modular-Avionics.html (2007).

[Feiler 2007b]
Feiler, Peter & Hansson, Jörgen. *Flow Latency Analysis with the Architecture Analysis and Design Language (AADL)* (CMU/SEI-2007-TN-010). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.

[Feiler 2008a]
Feiler, Peter H. & Hansson, Jörgen. "Impact of Runtime Architectures on Control System Stability." *Proceedings of 4th International Congress on Embedded Real-Time Systems*. Toulouse, France, 2008.

[Feiler 2008b]
Feiler, Peter H. & Hansson, Jörgen. "Impact of Runtime Architectures on Control System Stability." *Proceedings of 4th International Congress on Embedded Real-Time Systems*. Toulouse, France, 2008.

[Feiler 2008c]
Feiler, Peter H. "Efficient Embedded Runtime Systems through Port Communication Optimization." *Proceedings of 13th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS08), UML&AADL Workshop*. Belfast, Northern Ireland, 2008.

[Lewis 2008]
Lewis, Bruce A. & Feiler, Peter H. “Multi-Dimensional Model Based Development for Performance Critical Computer Systems Using the AADL.” *Proceedings of 4th International Congress on Embedded Real-Time Systems*. Toulouse, France, 2008.

Michotte 2008]
Michotte, Lydia; Feiler, Peter; France, Robert & Vergnaud, Vergnaud. “Aspect Oriented Modeling of Component Architectures Using AADL.” *Proceedings of Second International Conference on New Technologies, Mobility and Security*. Tangier, Morocco, 2008.

[Rugina 2008]
Rugina, Ana-Elena; Kanoun, Karama; Kaaniche, Mohamed & Feiler, Peter. “Software Dependability Modeling using an Industry-Standard Architecture Description Language.” *Proceedings of 4th International Congress on Embedded Real-Time Systems*. Toulouse, France, 2008.

3 Understanding the Relationship of Cost, Benefit, and Architecture

Ipek Ozkaya, Rick Kazman, and Mark Klein

3.1 Purpose

In this IRAD, we have investigated how architecture-level properties can be used to understand and control the costs and benefits of a system. Understanding structural properties of a software intensive system for economic analysis through code- or detailed design-level variables is often not possible. For example, the cost and benefit impact resulting from the integration of two systems, migration of functionality to a new system, or retiring a subsystem requires reasoning about properties that can only be inferred through architecture. Our goal is to provide guidance for practitioners to collect architecture-level metrics rather than rely only on code-level or detailed design-level effort and size information. We use the architecture-level metrics to give guidance to architects and managers in managing the costs and benefits of architectural change.

3.2 Background

Since software engineering artifacts exist to serve the business goals of an enterprise, optimizing the value of software systems is a central concern of software engineering [Boehm 2000]. It has become well recognized that quality attributes derive primarily from a system's software architecture. Consequently, quality attribute requirements are a driving force for architectural design [Bass 2003]. However, when cost and scheduling decisions are made for a software development project, the data used is often developer effort based on estimates of software size. Size estimates rely on very low level input, such as function points and/or estimates of source lines of code. While this important data helps forge a plan, many critical decisions need to be made early in the software life cycle-much earlier than when such estimates are typically available. These decisions revolve around how the technical approach aligns with the business goals and quality attributes for adding value to the organization. Currently, making value and cost decisions at the level of software architecture is difficult: clear understanding of architecture properties that can be analyzed with respect to costs and benefits does not exist and is currently done in an ad hoc fashion based largely on the experience and gut feelings of the architect.

Existing techniques for making cost and benefit judgments are grouped under algorithmic estimation, analogy-based estimation, and expert judgment [JPL 2003].

The most well-known algorithmic estimation model is Barry Boehm’s constructive cost model (COCOMO) [Boehm 1981]. Different variations of this suite of models have been developed with different focuses, for example the Constructive SoS Integration Cost Model (COSOSIMO), the Constructive System Engineering Cost Model (COSYSMO), and COCOMO II [COCOMO 2008]. COCOMO computes software development effort as a function of program size and a set of “cost drivers.” These models include assessments of product, hardware, personnel, and project attributes. Assessment of size is one of the key parameters, and it is either based on source lines of code (SLOC) () or function points, which are then adjusted for reuse. A set of 17 multiplicative effort multipliers and a set of 5 exponential scale factors are used. The model is calibrated using multiple regression analysis and a Bayesian approach [COCOMO 2008].

Function points are another widely known technique for algorithmic estimation. Developed in 1979 at IBM by Allen Albrecht as a language-independent technique for estimating project size, a function point is defined as one end-user business function, such as a query for an input. Function points takes the users' view and measure size based on screens, reports, and other external objects [Albrect 1983]. The quantifiable metrics are the number of external inputs, external outputs, external queries, internal logical files, and external interface files. The function point approach is used both as a method and also as a metric; size can be estimated by function points and input to other techniques, such as COCOMO. A function point maps easily into user-oriented requirements, but it also tends to hide internal functions, which also require resources to implement. The technique does not take into account any architectural concerns, such as infrastructure, integration, security.

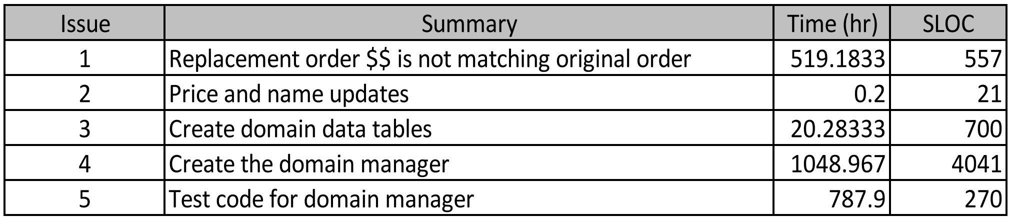
Analogy-based estimation was first proposed in 1977 by Sternberg [Keung 2007] and popularized in software engineering by Shepperd and Schofield in 1997 [Shepperd 1997]. The method is based on finding similar projects in historical portfolio data; hence, it utilizes case-based search, retrieval, and adaptation techniques heavily. The various project features used to determine project analogy differ widely in their relevance. A feature might represent a specific functional aspect such as *customer account management* and/or a project aspect such as *number of developers*. Features have varying impact on the analogies and, in turn, on the overall estimation accuracy and reliability. Existing approaches either try to find the dominant features or require experts to weight the features. The challenge lies in the ability to retrieve the most similar project and the magnitude of the historical data set. Considering too few projects may lead to unidentified projects, considering too many may lead to dilution of the closest analogies.

Analogy-based models do not seem as robust when using data external to the organization for which the model is built. In practice, expert judgment is the most widely used technique, which in fact is a form of analogy-based estimation but without the tool support. The method relies heavily on the ability to introduce historical data and the robustness of the algorithms in calculating the difference of the projects to find the closest match. The definitions of features can be flexible enough to include architectural aspects, although there have not been any such examples to date.

Our review of existing cost estimation techniques revealed the following:

- Existing cost estimation techniques attempt to analyze activities in a software development project in terms of the "optimum effort" to spend. A commonly cited example is Barry Boehm's analysis with COCOMO II for determining how much architecting is enough in a project [Boehm 2004]. The heuristic suggested by Boehm's study is that, for a 10 KSLOC project, more than 5 percent architecting investment unnecessarily increases schedule; for a 10,000 KSLOC project, one needs to spend about 40 percent effort in architecting; spending as little as 5 percent will double the development effort. These figures are generated using a combination of factors across projects such as risk resolution, source lines of code, and effort spent. Let us examine the 10,000 KSLOC project more closely. This analysis does not provide any direct insight about where that critical 40 percent effort should be spent, how the engineers should evaluate the effort they have spent on *productive* architecting, and how they can know what benefit they provide to the overall system effort. This study demonstrates that in large projects, in general, weak architecting causes large overruns. While such an analysis may justify the overall effort to be spent on architecting, it does not provide insight about how to justify one architecture over another in terms of costs and benefits. For example, a design that might reduce the integration cost, or modification costs down the road, over one that provides better run-time performance right away.

- Existing techniques used for estimating project costs and benefits based on software engineering artifacts rely heavily on size metrics. Size metrics alone do not provide architects with the tools to evaluate the impact of their decisions. Consider a real example in Figure 3-1. Issue 1 has 557 SLOC, issue 3 has 700, yet the time spent on issue 3 is significantly less. The issue summary hints at some content: in issue 3 new data tables are being created, whereas issue 1 suggests a problem (but does not provide any further insight). When examining the details of issue 1, we see that the problem is one of uncontrolled ripple effects. The architect and designers facing such a situation at best can only rely on their experience in the absence of architecture-level metrics to assist them in making such judgments.



Issue	Summary	Time (hr)	SLOC
1	Replacement order $$ is not matching original order	519.1833	557
2	Price and name updates	0.2	21
3	Create domain data tables	20.28333	700
4	Create the domain manager	1048.967	4041
5	Test code for domain manager	787.9	270

Figure 3-1: Sample SLOC and Time Data

- The creation of widely known cost estimation techniques date to early 1980s, predating the seminal work in software architectures in mid 1990s. The systematic omission of architectural considerations seems to be an outcome of the maturity of practice not catching up with architecture-level concerns.
- Controlling benefit does not appear as a concern in any of the cost estimation techniques. Earlier work at the SEI established the relationship of benefit, business goals, and quality attributes [Asundi 2001, Kazman 2002, Ozkaya 2007].

3.3 Approach

The approach we used in this study relied on both background search and empirical analysis. We analyzed data gathered from a development project that was going through architectural evolution. The data included architecture-level information as well as development and project management data. The background analysis relied on reviewing literature on existing cost estimation methods and analyzing how fit they were for considering cost and benefit at an architectural level.

3.4 Collaborations

The SEI participants in this study were Rick Kazman, Mark Klein, and Ipek Ozkaya. Collaborators included Prof. Mary Shaw of School of Computer Science, Carnegie Mellon

University, and Hakan Erdogmus of National Research Council of Canada. In addition, a team of students from the Master of Software Engineering (MSE) program in the School of Computer Science of Carnegie Mellon University worked on a prototype tool development. We also established industry collaborations with VistaPrint, Inc.

3.5 Evaluation Criteria

The a priori success criteria for judging the results of this IRAD were as follows:

- Identify architectural properties to assist in making cost and benefit analysis.
  - During this IRAD, we identified dependency and dependency-related metrics and their relationship to ripple effects as one key cost control property that can better be inferred at an architecture level. Coupling, and its relationship to architectural change, has long been established. Our approach used this as a primary factor in estimating cost and benefit for evolvability.
- Create requirements for economic-based reasoning tool support.
  - The MSE student team was able to produce a proof-of-concept prototype that we have started to use as part of a tutorial about economics-driven architecting.
- Create new techniques that take advantage of architectural information for controlling both costs and benefits.
  - We outlined an approach based on typical project data that one may get from an organization.
- Identify guidance for organizations about how to use architecture in thinking about cost and benefit.
  - We have collaborated with VistraPrint, Inc. where they put our approach to test.

We summarize these in our results section. Some of the early results were presented as part of an Economics-Driven Architecting tutorial at WICSA 2008 and OOPSLA 2008. Other results are being documented and will soon be submitted for peer review.

3.6 Results

3.6.1 Using Architecture to Think about Cost and Benefit

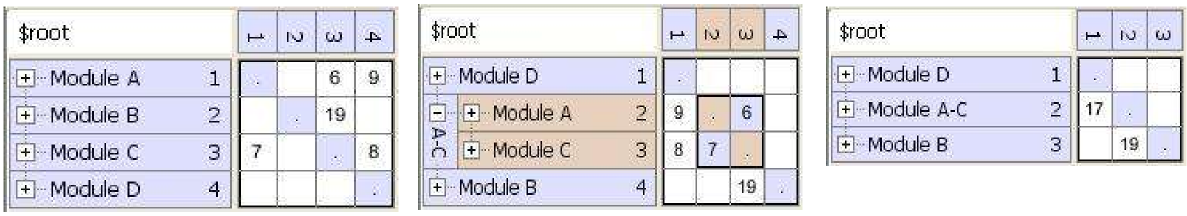
We identified dependency metrics as key to controlling cost and benefit via architecture. A typical cost estimation equation, common to the techniques we have reviewed, suggests the following

$$Effort = f(size)$$

Looking at architecture-level concerns suggests the following:

$$Effort = f(elements, communications, dependencies, ...)$$

Coupling metrics that can be inferred from dependency structure matrices (DSM) is a way of measuring architectural properties that affect change propagation and, consequently, cost. Benefit can derive from the ease of making modifications [Baldwin 2000]. Increased tool support can facilitate an understanding of dependencies by providing environments where both code and architectural views are present. Examples of these tools include Lattix and SonarJ [Lattix 2008, SonarJ 2008]. These tools also make this effort easier to quantify.



$root		1	2	3	4
⊞ Module A	1	.		6	9
⊞ Module B	2		.	19	
⊞ Module C	3	7		.	8
⊞ Module D	4				.

$root			1	2	3	4
⊞ Module D		1	.			
⊟ A-C	⊞ Module A	2	9	.	6	
	⊞ Module C	3	8	7	.	
⊞ Module B		4			19	.

$root		1	2	3
⊞ Module D	1	.		
⊞ Module A-C	2	17	.	
⊞ Module B	3		19	.

Figure 3-2: Reading dependencies with a DSM structure. Example from Lattix.

The numeric values in some of the cells in Figure 3-2 are *dependency strength*, defined as the total number of classes that each class in the source subsystem depends on in the target subsystem. The type of dependencies in a Java environment where element A is said to depend on element B can be any of the following: A inherits from B (implements for an interface); A calls a method or constructor in B; A refers to a data member in B; A refers to B (as in an argument in a method).

Using the dependency metric also leads to some lower-level metrics, although based on heuristics. For example, SonarJ suggests that an average *component dependency* metric should be kept and that normalized cumulative component dependency must not exceed 7 [SonarJ 2008]. Also, the architecture should not have any cycles of dependencies. These are architecture-level metrics and heuristics that assist in controlling cost.

We use dependency metrics along with typical project data that can be collected in the course of system development, such as tickets (work units), estimated SLOC, effort in time for each ticket, actual SLOC and effort in time for each ticket, and categories of tickets. Based on such data, we have defined an approach, outlined in Figure 3-3, for estimating the benefit of an architectural change. By collecting the information as shown, an architect or project manager can conduct architecture-level analysis where the impact of today's incurred cost, beyond merely counting SLOC, is revealed.

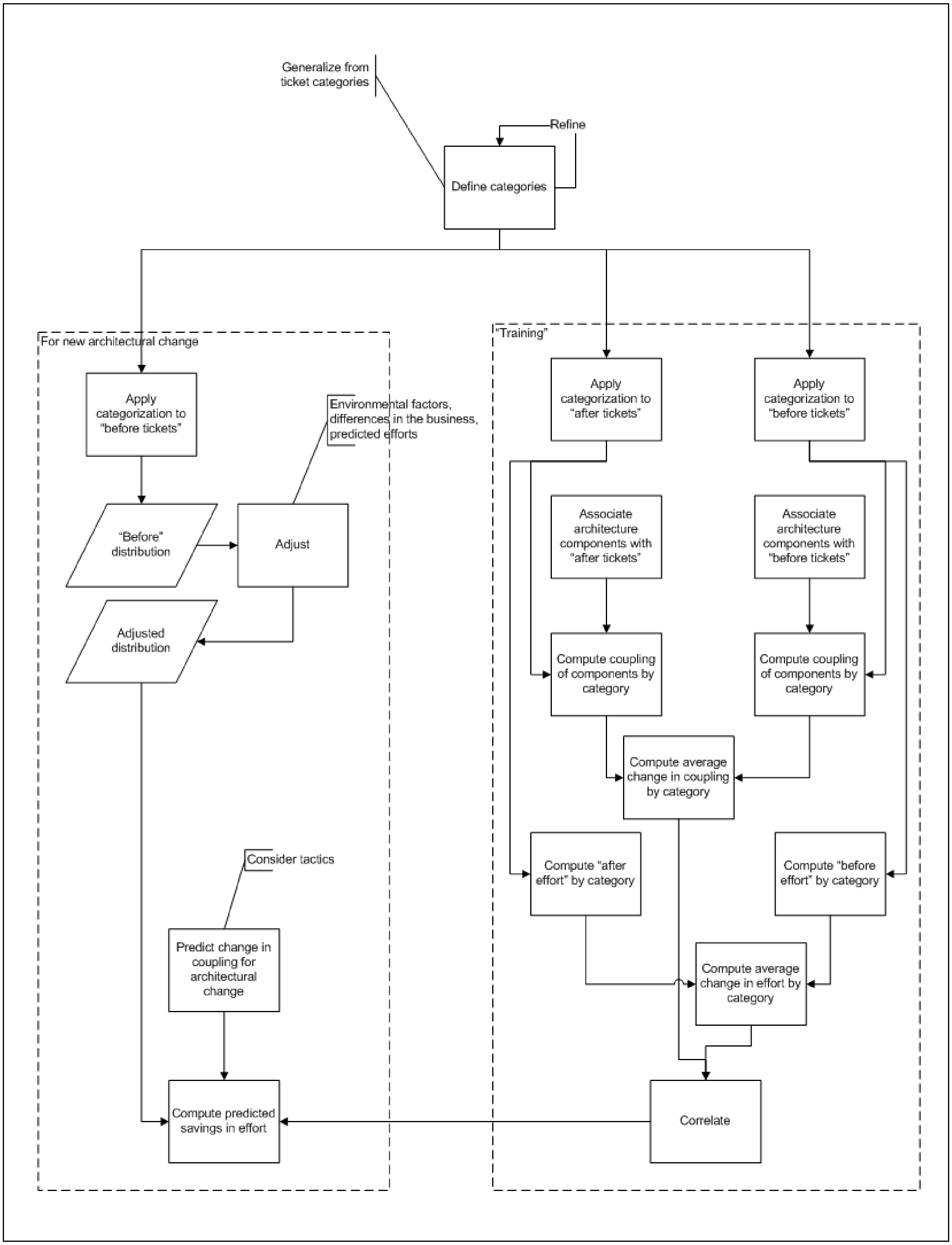

Generalize from ticket categories
Refine
Define categories
For new architectural change
Apply categorization to "before tickets"
Environmental factors, differences in the business, predicted efforts
"Before" distribution
Adjust
Adjusted distribution
Consider tactics
Predict change in coupling for architectural change
Compute predicted savings in effort
"Training"
Apply categorization to "after tickets"
Apply categorization to "before tickets"
Associate architecture components with "after tickets"
Associate architecture components with "before tickets"
Compute coupling of components by category
Compute coupling of components by category
Compute average change in coupling by category
Compute "after effort" by category
Compute "before effort" by category
Compute average change in effort by category
Correlate

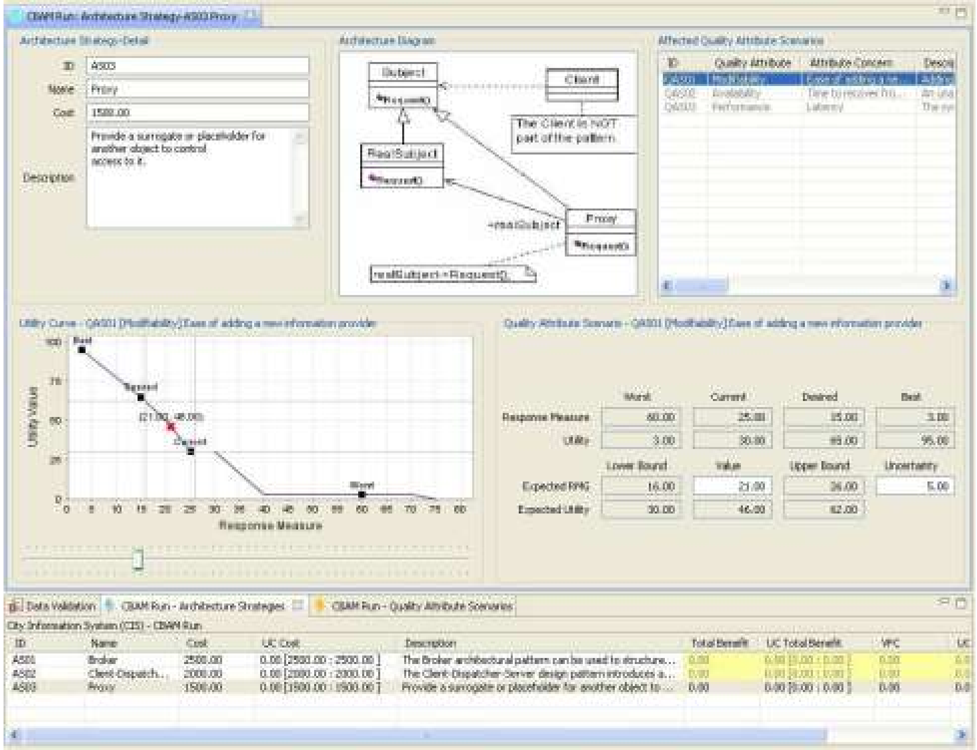
Figure 3-3: The approach followed in estimating coupling and its effect in cost

3.6.2 Tool Support for Economics-Driven Architecting

Another result of the IRAD is a prototype tool, EDA (Figure 3-4), for conducting economics-driven analyses for architecture design [EDA 2008]. Our empirical studies clearly revealed that this level of analysis requires what-if and sensitivity analyses, which are cumbersome to conduct without tool support.

The EDA tool, although only a prototype, provided the following benefits:

- EDA provided the ability to see economic information associated with quality attribute scenarios and architecture strategies.
- EDA provided the ability to manage many different categories of information. EDA created an opening for potentially connecting economic tools support with existing tools that allow for low-level data management (such as design structure matrices). Because the EDA tool uses the same environment as Lattix, a DSM tool, and ArchE [ArchE 2008], a logical next step is to see how they can be integrated.
- EDA facilitated the correlation of several classes of data. Usability is a key concern. To control cost and benefit through architecture there are several classes of data in several categories that need to be correlated, such as: quality attributes, architectural strategies, as-is architecture, business goals, economic concerns. The user needs to be able to run simulations. Without effective tool support knowing where to start can be challenging.


CBAM Run: Architecture Strategy-AS03 Proxy
Architecture Strategy-Detail
Architecture Diagram
Subject
Request()
Client
The Client is NOT part of the pattern.
RealSubject
Request()
Proxy
Request()
+realSubject
realSubject->Request();
Affected Quality Attribute Scenarios
Utility Curve - QAS01 [Modifiability] Ease of adding a new information provider
Best
Current
(21.00, 46.00)
Desired
Worst
Utility Value
Response Measure
0
25
50
75
100
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80
Quality Attribute Scenario - QAS01 [Modifiability] Ease of adding a new information provider
Data Validation
CBAM Run - Architecture Strategies
CBAM Run - Quality Attribute Scenarios

ID	AS03
Name	Proxy
Cost	1500.00
Description	Provide a surrogate or placeholder for another object to control access to it.

ID	Quality Attribute	Attribute Concern	Descr
QAS01	Modifiability	Ease of adding a ne...	Adding
QAS02	Availability	Time to recover fro...	An una
QAS03	Performance	Latency	The sy

	Worst	Current	Desired	Best
Response Measure	60.00	25.00	15.00	3.00
Utility	3.00	30.00	65.00	95.00
	Lower Bound	**Value**	**Upper Bound**	**Uncertainty**
Expected RMG	16.00	21.00	26.00	5.00
Expected Utility	30.00	46.00	62.00	

City Information System (CIS) - CBAM Run

ID	Name	Cost	UC Cost	Description	Total Benefit	UC Total Benefit	VFC	UC
AS01	Broker	2500.00	0.00 [2500.00 : 2500.00]	The Broker architectural pattern can be used to structure...	0.00	0.00 [0.00 : 0.00]	0.00	0.0
AS02	Client-Dispatch...	2000.00	0.00 [2000.00 : 2000.00]	The Client-Dispatcher-Server design pattern introduces a...	0.00	0.00 [0.00 : 0.00]	0.00	0.0
AS03	Proxy	1500.00	0.00 [1500.00 : 1500.00]	Provide a surrogate or placeholder for another object to ...	0.00	0.00 [0.00 : 0.00]	0.00	0.0

Figure 3-4: Economics-Driven Architecting Tool

Understanding cost and benefit structures through the architecture of a system allows making situated judgments earlier in a software project—before resources have been invested in suboptimal ways. Our investigations revealed that companies can have access to architecture-level information about their systems with low-key efforts and can also take advantage of their existing project management data for such analysis. Making such metrics available and integrating them allows an organization to better control cost. However, since architecture-level metrics also allow one to reason about quality, such metrics allow one to manage the *benefit* of an architecture as well.

3.7 References/Bibliography

URLs are valid as of the publication date of this document.

[Albrect 1983]
Albrecht, A.J. & Gaffney, J. R. "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation." *IEEE Transactions Software Engineering 9,* 6 (November 1983): 639-648.

[ArchE 2008]
SEI Architecture Expert Tool, 2008. http://www.sei.cmu.edu/architecture/arche.html

[Asundi 2001]
Asundi, Jayatirtha; Kazman, Rick & Klein, Mark. *Using Economic Considerations to Choose Among Architecture Design Alternatives* (CMU/SEI-2001-TR-035, ADA399151). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2001.
http://www.sei.cmu.edu/pub/documents/01.reports/pdf/01tr035.pdf

[Baldwin 2000]
Baldwin, C. Y. & Clark, K. B. *Design Rules: The Power of Modularity*. Cambridge, MA: MIT Press, 2000.

[Bass 2003]
Bass, Len; Clements, Paul & Kazman, Rick. *Software Architecture in Practice, Second Edition*. Boston, MA: Addison-Wesley Publishers, 2003 (ISBN: 0321154959).

[Boehm 1981]
Barry W. Boehm. *Software Engineering Economics*. Englewood Cliffs, N.J.: Prentice-Hall, (ISBN: 0138221227).

[Boehm 2000]
Boehm, B. & Sullivan, K.J. "Software economics: a Roadmap," *International Conference on Software Engineering (ICSE) 2000.* Limerick, Ireland, 2000.

[Boehm 2004]
Boehm, Barry W. & Turner, Richard. *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston: Addison-Wesley, 2004 (ISBN: 0321186125).

[COCOMO 2008]
Constructive Cost Model, 2008.
http://sunset.usc.edu/csse/research/COCOMOII/cocomo_main.html

[EDA 2008]
http://seiedatool.notlong.com

[JPL 2003]
Lum, K.; Bramble, M.; Hihn, J.; Hackney, J.; Khorrami, M. & Monson, E. "Handbook for Cost Estimation." JPL Report, 2003. http://ceh.nasa.gov/downloadfiles/Web%20Links/cost_hb_public-6-5.pdf

[Kazman 2002]
Kazman, Rick; Asundi, Jai & Klein, Mark. *Making Architecture Design Decisions: An Economic Approach* (CMU/SEI-2002-TR-035, ADA408740). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002.
http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr035.pdf

[Keung 2007]
Keung, J. & Kitchenham, B. "Optimizing Project Feature Weights for Analogy-based Software Cost Estimation using Mantel Correlation." *14th Asia-Pacific Software Engineering Conference: Proceedings*. Nagoya, Aichi, Japan, 2007.

[Lattix 2008]
Lattix – Software for Architecture Management. http://www.lattix.com/ (2008).

[Ozkaya 2007]
Ozkaya, Ipek; Kazman, Rick & Klein, Mark. *Quality-Attribute-Based Economic Valuation of Architectural Patterns* (CMU/SEI-2007-TR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.
http://www.sei.cmu.edu/pub/documents/07.reports/07tr003.pdf

[Shepper 1997]
Shepperd, M. & Schofield, C. "Estimating Software Project Effort Using Analogies." *IEEE Transactions on Software Engineering 23,* 12 (November 1997): 732-743.

[SonarJ 2008]
http://www.hello2morrow.com/products/sonarj/whitepapers (2008).

4 Mechanism Design

Daniel Plakosh, Mark Klein, Gabriel Moreno, and Kurt Wallnau

4.1 Purpose

All computational systems have resource limitations, for example processing, memory, and network capacity. Combat systems are no exception. Future combat systems are also no exception: they may have significantly more computational resources than current generation combat systems, but these will surely be consumed by new war fighting doctrine. Providing efficient resource management in combat systems has always been, and will remain, a challenging engineering problem, the more so because the meaning of “efficient” can be elusive.

Centralized resource allocation becomes problematic as systems grow in scale and complexity. A centralized decision maker must know what is needed at any time by all the parts of a system, including its “user parts.” At some point, the diversity and number of tasks that a system must perform makes this kind of omniscience impossible. Economic markets are a distributed solution to resource allocation—efficient allocation decisions are based on information provided by the participants in the market.

The research described here addresses the question, “Can economic mechanisms be used to allocate scarce computational resources in DoD systems?” Our proximate interest is in tactical systems since these pose demanding and DoD-unique requirements. As discussed later, there is already ample evidence that economic mechanisms have a role to play in DoD e-commerce applications such as supply chain management.

4.2 Background

A *mechanism* is an institution such as an auction, voting protocol, or a market that defines the rules or protocols governing how *rational* and *intelligent agents* interact, and how collective decisions are made. In a *computational mechanism*, agents are computational processes that work on behalf of rational and intelligent humans. *Mechanism design* is a sub-discipline of economics and game theory, and is the art and science of designing mechanisms that achieve prescribed and desirable global outcomes.

Each term in the phrase “rational and intelligent” is crucial and has special meaning in the context of mechanism design. A *rational* agent seeks to maximize its outcome. Because the technical language of microeconomics is often expressed in terms of economic utility, a rational agent is also often referred to as a utility maximizer. An *intelligent* agent knows as much about the workings of a mechanism as does the mechanism designer. Colloquially, a rational and intelligent agent is one that exhibits self interest and guile.

Recall the earlier assertion that centralized resource allocation fails at some (admittedly difficult to define) threshold of system scale and complexity. At this threshold, the allocation function does not possess sufficient information to choose an optimal resource allocation, or, more technically, to solve some arbitrarily complex resource optimization problem. Instead, the allocator must obtain information from the distributed parts of the system. If, for example, the parts of the system are human operators with a stake in the outcome of an allocation decision, it is fair to ask whether they have an incentive to provide truthful information to the allocator, or whether instead they might have an incentive to be deceptive, if deception would produce a "better" allocation for the operators.

The issue of human incentives is therefore central to mechanism design, and it is this aspect of the research that is of distinct value to automated decision processes such as bandwidth allocation. We can safely assume that humans will behave in a rational and intelligent way—with self interest and guile—if doing so will be to their benefit. On the other hand, this need not be regarded as an evil to be rectified. Indeed, market systems work precisely because they exploit the virtues of selfishness, and thus the ability of each part of the system to maximize its local utility.

The "trick" of mechanism design is to define a decision-making institution that aligns the incentives of the participants in the institution with those of the mechanism designer, so that a participant in the mechanism maximizes its own (local) utility by behaving in a way that maximizes the system (global) utility. This trick is often, but not always, achieved through the use of a payment scheme that compensates for "right" behavior and taxes for "wrong" behavior so that, as a consequence, rational and intelligent participants always behave in the right (desired) way.

A more formal, though still quite general, description of the trick is to regard the task of the mechanism designer as one that makes a (for instance) allocation decision that maximizes global efficiency, sometimes called a social choice function:

$$F = \max \sum_i v_i$$

Eq 4-1

where v_i is the value v of some decision outcome for agent i. Now, each participant in the mechanism is seeking to maximize its own utility function u. Its utility for some outcome i is defined as:

$$u_i = {}'_i -$$

Eq 4-2

where v_i is the *intrinsic value* the participant has for outcome *i,* say a particular allotment of network bandwidth, and t_i is the mechanism induced payment for that bandwidth allocation. The art of mechanism design is to find an appropriate t such that each participant maximizes its own utility by truthfully reporting v. In this case, the mechanism achieves (or computes) the social choice function.

It is worth observing that finding the right payment mechanism is not a trivial undertaking, not the least because it requires that the mechanism designer understand the nature of value for each participant.

Mechanism design has a long and storied research tradition, most recently exhibited in the awarding of three Nobel prizes in 2007 for work in laying the foundations of mechanism design.[16] Mechanism design is not only of academic interest, of course. The US Federal Communication Commission (FCC) routinely conducts auctions for radio spectra, each of which must be meticulously designed to achieve prescribed social objectives in the face of stringent competitive interests.[17] Computational mechanism design has already had a considerable impact on the US economy, most famously Google's ad placement auction accounted in 2006 to more than 98% of their total revenues.[18]

The use of computational mechanisms for allocating scarce computational resources is comparatively less well developed, although it has emerged as an active area of research, with examples that include mechanisms for allocating processor cycles for scientific computing on the worldwide grid, for routing network packets, for allocating network capacity, for allocating tasks to autonomous robots, and (most pertinent to the work reported here) for fusing sensor data.[19]

4.3 Approach

Our investigation focused on the use of economic mechanisms to achieve an efficient allocation of network bandwidth for a tactical data network. We developed prototype software to assess the practicality of using computational mechanisms in demanding, DoD-unique settings.

In the first year of the investigation, we developed a realistic emulation of a tactical data network modeled on LINK-11. In the second year, we retargeted the basic mechanism to a more heterogeneous, but still experimental, peer-to-peer tactical network being implemented on the Cursor-On-Target (CoT) router.[20] Table 4-1summarizes the key complementary features of these mechanisms.

LINK-11 Sensor Fusion	CoT Bandwidth Allocation
Homogenous agents: ships (platforms) with radar	Heterogeneous agents: surveillance, security, intelligence, etc.
Homogeneous data: radar tracks	Heterogeneous data: UAV video, biometric, etc.
Homogeneous value: radar quality defined as range and bearing covariance	Heterogeneous value: agent type-specific value expressed in an executable doctrine language
Broadcast communication on shared radio	Unicast and peer-to-peer communication

Table 4-1: High-Level Comparison of Prototype Tactical Network Mechanisms

[16] See http://nobelprize.org/nobel_prizes/economics/laureates/2007 for details.

[17] See http://wireless.fcc.gov/auctions/default.htm?job=auctions_home for details.

[18] B Edelman, M Ostrovsky, M Schwarz, "Internet Advertising and the Generalized Second Price Auction: Selling Billions of Dollars Worth of Keywords," American Economic Review, 2007.

[19] See http://www.sei.cmu.edu/publications/documents/08.reports/08tr004.html for examples.

[20] See http://www.mitre.org/news/the_edge/summer_07/robbins.html for details.

4.4 LINK-11 Bandwidth Allocation for Radar Sensor Fusion

LINK-11 is a collection of digital data link protocols for communications among a number of participating units. Communication on the link takes place by round robin, designated roll call. Each unit reports when requested to do so by a participating unit that has been designated as Net Control Station.

At 2250 BPS for data (a bit more for voice), network bandwidth is a scarce resource in LINK-11. Even its successor, LINK-16, has only 28.8 KBS for data. To conserve bandwidth, LINK-11 uses a reporting responsibility ("R2") protocol where exactly one platform assumes R2 for each radar contact, and only this platform reports data for that contact. While this approach has the virtue of conserving bandwidth, it sacrifices opportunities to fuse track data to improve the quality of the common operating picture.

We auction additional quanta of bandwidth, and allow the participating units themselves to decide which track data will be most valuable. For this purpose, we used the Vickrey-Clarke-Groves (VCG) mechanism as our starting point. The VCG auction is a generalization of the second-price sealed bid auction.[21] The VCG auction has the key property of "incentive compatibility," which ensures that each participating unit will maximize their payoff only by truthfully revealing its private information.

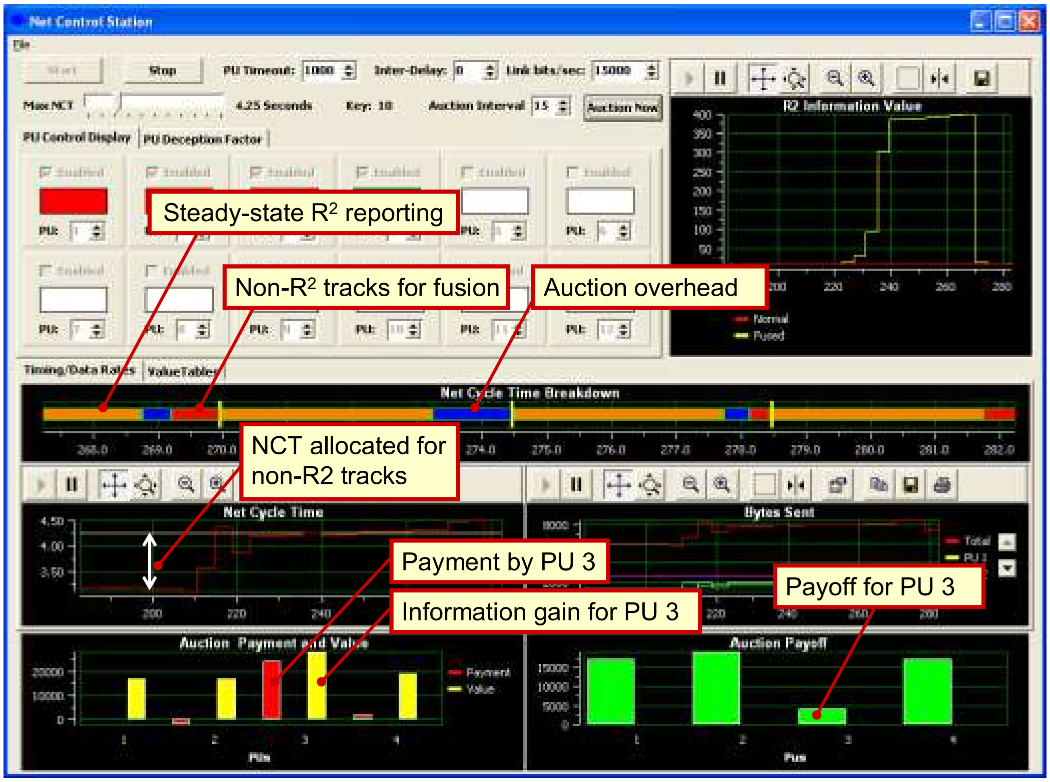
The incentivized payoff structure for the bandwidth auction is defined in Eq 4-3, which reflects each participant's marginal contribution to total information gain, where u_i and v_i are payoff and value functions for participant i, respectively; Z is the information that participant i has for all tracks; F^* and F_{-i}^* is the optimal bandwidth allocation with and without participant i included in the auction, respectively.

$$u_i(Z,F^*) = v_i(Z,F^*) - \left[\sum_{j \neq i} v_j(Z,F_{-i}^*) - \sum_{j \neq i} v_j(Z,F^*) \right]$$

Eq 4-3

The payoff structure in Eq 4-3 incentivizes each participant to maximize its payoff; and, its payoff is maximized by maximizing the information gain of the whole group. A comparison with Eq 4-2 shows that the term to the right of the equals sign is the value of an allocation to a platform while the bracketed term in Eq 4-3is the payment.

[21] Bids are secret and the winner pays second-highest bidder's bid.


Net Control Station

PU Timeout: 1000 Inter-Delay: 0 Link bits/sec: 15000

Max NCT 4.25 Seconds Key: 10 Auction Interval 15

Figure 4-1: Screenshot of Network Control Station Auction Monitor

Figure 4-1 depicts a snapshot of the auction at runtime. The vertical, two-headed arrow labeled “NCT allocated for non-R2 tracks” shows the quantum of bandwidth auctioned for the purpose of data fusion. The horizontal bar labeled “Steady state R2 reporting” shows how bandwidth is used, and includes the cost of running the auction itself. The auction is run periodically, for example once every fifteen network cycles. The economic outcome for one auction is shown at the bottom of the figure. In this example, participating unit 3 (PU 3) gains the most information but also makes the largest payment. PU 3’s payment represents its adverse impact on the other participants. That is, if PU 3 were not in the auction, its payment (red bar) would be distributed as information gain (yellow bar) among the remaining participants.

Table 4-2 provides a high level summary of the LINK-11 auction mechanism in terms of seven recurring themes of a mechanism solution to a resource allocation problem.

Mechanism Consideration	Mapping to LINK-11
Scarce resource	Bandwidth for track data
Participants	Platforms (ships)
Social choice function	Maximize total information gain for added net cycle time
Private information	Track quality, expressed as range and bearing covariance
Intrinsic value	Contribution to information gain is “cashed in” after action
Rules	Single-shot, sealed-bid, Vickrey-Clarke-Groves payoff
Currency	Information gain (reduced range and bearing covariance)

Table 4-2: Summary of LINK-11 Sensor Fusion Auction Mechanism

4.5 Cursor on Target Mechanism

The Cursor-on-Target (CoT) mechanism was developed in the second and final year of the investigation. VCG payoff rules are used for both LINK-11 and VCG mechanisms, but the similarity between these mechanism ends there. Our objective in developing the CoT mechanism was to demonstrate that analogous economic mechanisms can be used to allocate tactical network bandwidth in substantially different tactical environments, both with respect to infrastructure (network) technology and doctrine.

The experimental environment used to demonstrate the Cot auction mechanism is known as the Tactical Network Topology (TNT) testbed. TNT is a quarterly, cooperative field experiment hosted by the Naval Postgraduate School and conducted by U.S. Special Operations Command, with simultaneous demonstrations of new technologies between three locations: Camp Dawson, West Virginia., Camp Roberts, California, and Camp Atterbury, Indiana. The series of quarterly field experiments is intended to demonstrate technologies that bridge "the information gap in the 'Last Tactical Mile,' headquarters to tactical units in remote locations." [22]

The general objective for the CoT auction mechanism was to demonstrate innovative bandwidth adaptation solutions that dynamically adjust both application load and network configuration to achieve high service-level quality as defined by the user's stated "value" of a service. Specifically, by applying an auction mechanism to the service-level requirements as provided by network users, the mechanism can alter both the application traffic (via throttling) and the location of mobile network nodes to effect greater available bandwidth, and thus service quality, to high-valued services.

The auction experiment conducted August 21, 2008 demonstrated value functions for bandwidth allocation in an air-ground sensor network that used multiple unmanned air vehicles (UAVs) and ground sensors. These assets were providing discrete data via CoT messages; namely, live position reports, video via rapid still-frame imagery, and sensor alerts. Users in the network operations center (NOC) and tactical operation center (TOC) were able to state their role-specific information requirements, which were processed by the auction mechanism and informed the adjustment of resources toward or against specific data. These adjustments were effected either by the management of specific CoT flows or by the adjustment of the physical network topology.

Figure 4-2 shows the NOC during a scenario where there were four different role-specific views.

In the LINK-11 mechanism, the value of information had a precise meaning and an associated currency: information gain resulting from sensor data fusion, expressed in terms of covariance of sensor error on range and bearing. There was also just one kind of participant in the auction: platforms (ships). Explicit rules of interaction among platforms were also established by the LINK-11 protocol: round-robin and roll-call. Collectively, we will refer to these (somewhat imprecisely) as "doctrine." In contrast, the tactical environment exhibited in the TNT experiments lacks any fixed doctrine. This is due in part to the thematic emphasis on experimental technologies at TNT. However, it is also no doubt a reflection of the diversity in the types of missions undertaken by special operations forces.

[22] Excerpted from http://www.biometrics.dod.mil/Newsletter/Issues/2008/July/v4issue3/v4Issue3_a2.html

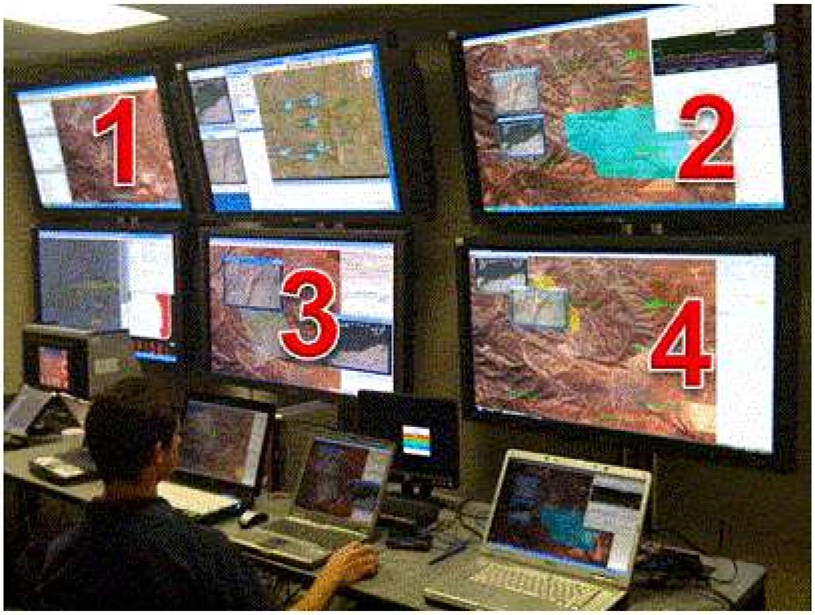
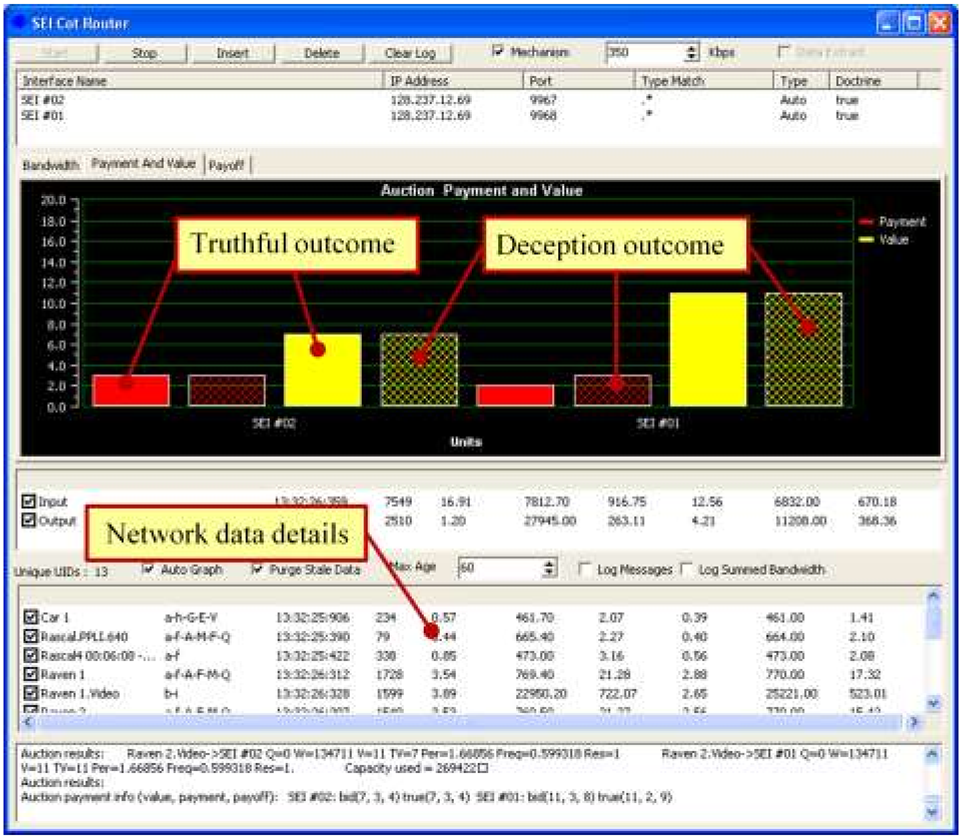


Figure 4-2: Role-Specific Tactical Displays (1-4) in TNT Network Operation Center

Accommodating diverse doctrinal requirements requires a generalized scenario framework from which mission-specific scenarios can be constructed. In the following three-part scenario framework, we denote participants in a tactical network as operators, and there is a designated operator called the commander.

1. Operators realize intrinsic value by doing their best to fulfill roles in a mission. Each operator is trained to fulfill a number of roles (e.g., forward observer, HQ security, or reconnaissance) for any given mission. Each role will have its own notions of intrinsic value. This value is imparted to operators during their training as “doctrine.”
2. Mission-specific value functions can be obtained in a variety of ways. The commander might assign each operator a value function. Or, operators might create their own value functions based only on general commander instructions. Also possible, operators might create value functions as mission-specific instances of generic value function imparted during training.
3. Bandwidth is a scarce resource that is needed for operators to fulfill their roles. Operators swill act strategically (i.e., with guile) to ensure sufficient allocation to maximize their utility.

Besides a general scenario framework, the mechanism requires a concrete way of expressing role-specific doctrine. The prototype CoT mechanism encodes value functions in an executable subset of the “C” programming language. This is the private information transmitted by operators to the auctioneer.



☑ Mechanism 350 Kbps ☐ Show Extract

Interface Name	IP Address	Port	Type Match	Type	Doctrine
SEI #02	128.237.12.69	9967	.*	Auto	true
SEI #01	128.237.12.69	9968	.*	Auto	true

☑ Input		13:32:26:359	7549	16.91	7812.70	916.75	12.56	6832.00	670.18
☑ Output			2510	1.20	27945.00	263.11	4.21	11208.00	368.36

Unique UIDs : 13 ☑ Auto Graph ☑ Purge Stale Data Max Age 60 ☐ Log Messages ☐ Log Summed Bandwidth

☑ Car 1	a-h-G-E-V	13:32:25:906	234	0.57	461.70	2.07	0.39	461.00	1.41
☑ Rascal.PPLI.640	a-f-A-M-F-Q	13:32:25:390	79	.44	665.40	2.27	0.40	664.00	2.10
☑ Rascal4 00:06:00 -...	a-f	13:32:25:422	338	0.85	473.00	3.16	0.56	473.00	2.08
☑ Raven 1	a-f-A-F-M-Q	13:32:26:312	1728	3.54	769.40	21.28	2.88	770.00	17.32
☑ Raven 1.Video	b-i	13:32:26:328	1599	3.09	22950.20	722.07	2.65	25221.00	523.01

```
Auction results:    Raven 2.Video->SEI #02 Q=0 W=134711 V=11 TV=7 Per=1.66856 Freq=0.599318 Res=1        Raven 2.Video->SEI #01 Q=0 W=134711
V=11 TV=11 Per=1.66856 Freq=0.599318 Res=1.           Capacity used = 269422□
Auction results:
Auction payment info (value, payment, payoff):   SEI #02: bid(7, 3, 4) true(7, 3, 4)  SEI #01: bid(11, 3, 8) true(11, 2, 9)
```

Figure 4-3: Snapshot of the CoT Auction Showing Truthful and Deception Payoffs

Analogous to the LINK-11 auction mechanism, the prototype CoT auction mechanism includes infrastructure to study the outcomes of auction allocations from journal logs and in real time. Both prototype mechanisms also include the ability to assess the effect of deception or other forms of attempted strategic manipulation by operators (or platforms in the LINK-11 case). Figure 4-3 is a snapshot of the auction where there are two participants (for simplicity of presentation), and where one of the participants (SEI #01) has provided a deceptive evaluation of its value.[23] As expected, the payoff under deception (value minus payment) is worse than it would be for truth telling: here, SEI #01 makes a higher payment for its bandwidth allocation when lying.

Table 4-3 provides a high-level summary of the CoT auction mechanism in terms of seven recurring themes of a mechanism solution to a resource allocation problem.

[23] To facilitate our study we explicitly annotate deceptive behavior so that dual auctions can be run. Obviously an operator would not announce a deceptive response; in a deployed system the operator would simply receive the deception payoff.

Mechanism Consideration	Mapping to LINK-11
Scarce resource	Bandwidth for mission data
Participants	Agents fulfilling different mission roles
Social choice function	Maximize total role-specific, doctrine-specified values
Private information	Local assessment of information requirements
Intrinsic value	“Utils” are cashed in after action.
Rules	Single-shot, sealed-bid, Vickrey-Clarke-Groves payoff
Currency	“Utils”

Table 4-3: Summary of CoT Bandwidth Allocation Auction Mechanism

4.6 Collaborations

The prototype CoT mechanism was developed collaboratively with colleagues from Harvard University (Dr. David Parkes and Sven Seuken) and the Naval Postgraduate School (NPS) (Dr. Alex Bordetsky and Michael Clements). NPS provided domain expertise in a specific tactical network setting, and provided an environment for conducting more demanding field experiments with bandwidth auction mechanisms.

4.7 Evaluation Criteria

Although mechanism design and computational mechanism design remain active areas of fundamental research, their theoretical foundations are well established. Our evaluation criteria therefore centered on issues of practicality:

1. Do the mathematical tools and concepts of computational mechanism design (microeconomics, game theory) have relevance to tactical systems of interest to the DoD?
2. Can computational mechanisms work in demanding (high-availability, real-time, resource-constrained) settings?
3. Is computational mechanism design susceptible to routine engineering design, or does it require substantial, rarified skills? Is mechanism engineering a discipline whose time has come?

4.8 Results

Expressed in terms of our evaluation criteria, our results indicate the following:

1. Computational mechanism design provides a new and useful set of design tools for DoD systems, especially those that are “net centric” and for which some degree of continuous, real-time self-adaptation of the system to changing mission needs is desired.
2. Our prototype implementations suggest that computational mechanisms can indeed work in the kinds of demanding system contexts presented by DoD tactical systems. The LINK-11 mechanism uses LINK-11 algorithms, and in some cases code, as an experimental platform; the CoT mechanism has been deployed in two quarterly field demonstrations.

3. It is not yet clear that computational mechanism design can be made a matter of routine engineering practice. While foundations such as the well-studied VCG auction are available, issues of doctrine and incentives in combat systems are still rather obscure, and these issues invariably lie at the heart of any mechanism design problem.

While we cannot yet affirm that computational mechanism design is practical as a broad-based solution to resource management challenges in tactical systems, we have established a sufficient basis to conclude that these techniques may be applicable to some DoD systems. Moreover, we believe the language of mechanism design brings to the fore issues of human value and incentives that must be addressed in net-centric systems, especially those that are predicated on pushing decision making to the edges of a system (i.e., to individual operators and to the "Army of One").

4.9 Publications and Presentations

[Klein 2008]
Klein, Mark; Plakosh, Daniel & Kurt Wallnau. *Using the Vickrey-Clarke-Groves Auction Mechanism for Enhanced Bandwidth Allocation in Tactical Data* (CMU/SEI-2008-TR-004). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2008.

[Klein 2008b]
Klein, Mark; Moreno, Gabriel; Parkes, David; Plakosh, Daniel; Seuken, Sven & Kurt Wallnau. "Handling Interdependent Values in an Auction Mechanism for Enhanced Bandwidth Allocation in Tactical Data Networks." *Proceedings of the 2008 Workshop on the Economics of Networks, Systems, and Computation*. Seattle, WA, 2008.

[Klein 2008c]
Klein, Mark; Plakosh, Daniel & Wallnau, Kurt. "Mechanism Design for Sensor Fusion: Tactical Networks as a Foil for Ultra-Large Scale Systems." *Proceedings of the 2nd International Workshop on Ultra-Large Scale Systems ULSSS'08*. Leipzig, Germany, 2008.

[Klein 2008d] (Presentation)
Klein, Mark; Moreno, Gabriel; Plakosh, Daniel & Wallnau, Kurt. "Using Vickrey-Clarke-Groves Auctions to Allocate Bandwidth for Sensor Fusion in Tactical Data Networks." *National Defense Industrial Association (NDIA) 9th Annual Science & Engineering Technology Conf DoD/Tech Exposition*. North Charleston, SC, 2008.

[Plakosh 2008] (Presentation, ½-day Tutorial)
Plakosh, Daniel and Wallnau, Kurt. "Practical Computational Mechanism Design." Pittsburgh, PA: Robert Morris University, 2008.

[Wallnau 2009] (Presentation, accepted)
Wallnau, Kurt. "Economic Mechanisms for Allocating Tactical Network Bandwidth." *Systems and Software Technology Conference (SSTC 2009)*. 2009.

5 Assurance Cases for Medical Devices

John Goodenough, Charles B. Weinstock, Paul Jones, Prof Insup Lee and Sherman Eagles

5.1 Purpose

The medical device industry finds itself moving inexorably in the direction of so many other industries—an ever-increasing percentage of device functionality is provided by software. The industry is beginning to experience the kinds of problems that arise when products that were formerly mostly hardware become significantly dependent on software for their safe and effective operation. In particular, the increasing complexity of medical device software raises new questions about how manufacturers and regulators are to gain confidence in the safe operation of such software.

While there are many similarities between medical devices and other systems for which software safety is important, the combination of privacy concerns and the regulatory environment associated with medical devices makes the assurance problem more complex. Adding to these difficulties is the desire for “plug-and-play” use of these devices in a hospital environment. The payoff of plug-and-play medical devices is potentially large but, as with all such systems of systems, there is the potential for potentially unsafe or insecure emergent behavior.

The current practice for ensuring safety is processed-focused—relying mostly on evaluating compliance with safety regulations and standards. Food and Drug Administration (FDA) personnel and some medical device manufacturers have indicated in discussions with us an interest in more product-focused device assurance practices, allowing manufacturers to focus safety assurance efforts more on demonstrations of device safety rather than on gathering indirect data supporting the soundness of their design and production practices. Because assurance cases are product focused, the FDA and some manufacturers are considering their use as a means of gaining more confidence in the safety of medical devices and in expediting the certification/approval process.

5.2 Background

The Software Engineering Institute first began considering the assurance case as a method of software assurance in 2004. Since then, interest in the technique has become widespread. An international community, including the SEI, has been researching the issues involved with developing security assurance cases and has held several workshops on the subject. The Object Management Group (OMG) has established a working group in the area,[24] and the International Organization for Standardization (ISO) is considering a standard (15026) that includes assurance cases. The U.S. Department of Homeland Security’s Build Security In website contains discussions regarding the use of assurance cases.[25]

[24] http://swa.omg.org

[25] https://buildsecurityin.us-cert.gov/daisy/bsi/articles/knowledge/assurance.html

The SEI took up the subject of assurance cases with the FDA in 2005-2006. By late 2006, AdvaMed, an advocacy group for the medical device industry, became interested in the subject and invited us to talk to them in early 2007. Since then, it has held a workshop on the subject and is considering additional activities. Other workshops have been held by the University of Pennsylvania, Massachusetts General Hospital, the University of Minnesota, and the FDA. Two independent projects have spun out of manufacturer and FDA interest in exploring product-focused assurance: a pacemaker "grand challenge" project,[26] based on information provided by a device manufacturer, and a NSF-sponsored project to specify and assure a generic infusion pump, conducted by the University of Pennsylvania and the FDA [Arney 2008]. The purpose of the pacemaker project is to challenge the assurance community to come up with a formally assured pacemaker design and breadboard implementation. The purpose of the generic infusion pump project is to develop a reference design for infusion pumps. This reference design will be used to research issues in verification and validation of embedded systems and will also be useful for manufacturers of infusion pump devices. We adopted this design as input to this IRAD project.

5.3 Approach

We initially planned to use a design for a Generic Infusion Pump (GIP) developed by Insup Lee and his team at the University of Pennsylvania, with U.S. Food and Drug Administration assistance, to explore the issues involved in using assurance cases in the medical device industry. We ended up using it (and the associated requirements and hazard analysis) as inspiration for the (imaginary) pump employed in this case. We're using this pump rather than a real infusion pump to avoid proprietary issues that might restrict the usefulness and distribution of our results. We've assumed that the pump we are assuring includes a complete drug library and barcode readers.

In constructing this assurance case, we were guided by current FDA thinking regarding the technical issues that must be addressed to ensure the safety of a medical device. We wanted to determine what would be useful to the FDA and what would be useful to the device manufacturer, then determine the best way to fit an assurance case practice into the development and approval of medical devices.

5.4 Collaborations

The SEI project team included John Goodenough and Chuck Weinstock. We also received valuable assistance from Austin Montgomery.

We were assisted by Paul L. Jones, Brian Fitzgerald, Rick Chapman, and Rauol Jetley at the U.S. Food and Drug Administration; Sherman Eagles and Patti Krantz at Medtronic (a device manufacturer); Insup Lee, Oleg Sokolsky, and David Arney at the University of Pennsylvania; and Andrew Urbach and Donna Flook at the Children's Hospital of Pittsburgh. All of these individuals contributed their time without compensation from the SEI.

[26] http://sqrl.mcmaster.ca/pacemaker.htm

5.5 Evaluation Criteria

At the beginning of this work, we determined at least some of the following would need to happen for the project to be deemed a success:

- We are able to establish significant and long lasting relationships with industry and FDA players in the medical device community.
- Manufacturers begin to come to the SEI for advice on how to structure, document, and use a medical device safety case.
- Manufacturers begin to come to the SEI to learn how to implement their own practices for developing medical device safety cases.
- The FDA begins to adopt some of the recommendations from our jointly created report.

We believe we have met or exceeded our a priori expectations in most of the above criteria.

5.6 Results

5.6.1 An Introduction to the Goal Structured Assurance Case

An assurance case is somewhat similar to a legal case. In a legal case, there are two basic elements. The first is evidence, be it witnesses, fingerprints, DNA, etc. The second is an argument given by the attorneys as to why the jury should believe that the evidence supports (or does not support) the claim that the defendant is guilty (or innocent). A jury presented with only an argument that the defendant is guilty, with no evidence that supported that argument, would certainly have reasonable doubts about the guilt of the defendant. A jury presented with evidence without an argument explaining why the evidence was relevant would have difficulty deciding how the evidence relates to the defendant.

The goal-structured assurance case is similar. There is evidence that a property of interest (i.e., safety) holds. For instance, there might be test results collected into a report. Without an argument as to why the test results support the claim of safety, an interested party could have difficulty seeing its relevance or sufficiency. So a goal-structured assurance case specifies a claim regarding a property of interest, evidence that supports that claim, and provides a detailed argument explaining how the evidence supports the claim.

In our case, the top-level claim is “The Generic Infusion Pump (GIP) is safe.” From that claim flows an argument that supports the top-level claim. The argument consists of one or more subsidiary claims that, taken together, make the top-level claim believable. These lower-level claims are themselves supported by additional claims until, finally, a sub-claim is to be believed because evidence exists that clearly shows the sub-claim to be true.

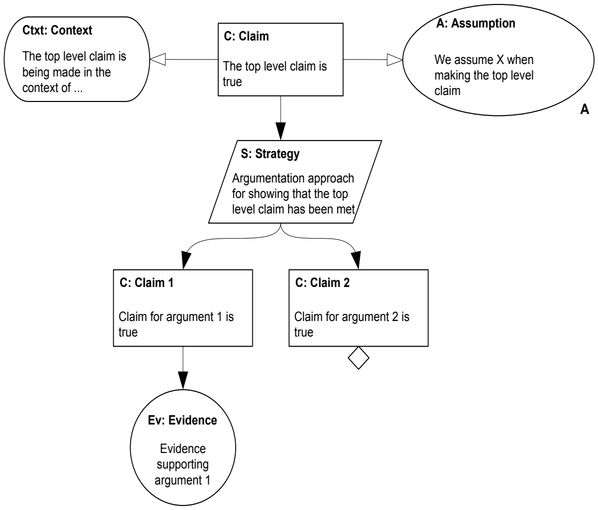

Ctxt: Context
The top level claim is being made in the context of ...
C: Claim
The top level claim is true
A: Assumption
We assume X when making the top level claim
A
S: Strategy
Argumentation approach for showing that the top level claim has been met
C: Claim 1
Claim for argument 1 is true
C: Claim 2
Claim for argument 2 is true
Ev: Evidence
Evidence supporting argument 1

Figure 5-1: Example GSN Argument

To develop the GIP assurance case and make it reviewable by others, we adopted the Goal Structuring Notation (GSN) developed by Tim Kelly and his colleagues at the University of York in the United Kingdom [Kelly 1998]. This notation has been used successfully in many safety cases and is ideally suited for our work. Figure 5-1 shows a short assurance case developed in GSN. In it, the top-level claim is labeled “Claim,” the argument consists of the sub-claims: “Claim 1” (supported by some evidence) and “Claim 2.” Other elements shown in the sample are the diamond under “Claim 2,” which indicates that the claim requires further development; the parallelogram labeled “Strategy,” which is meant to be a guide to the reader as to how the argument is structured; a rounded rectangle labeled “Context” and an oval with an “A” under it labeled “Assumption,” both of which provide explanatory information about the claim to which they are attached.

These are by no means the only elements to a goal-structured assurance case, but they represent those used in the GIP assurance case that follows later in this note.

5.6.2 Developing the GIP Assurance Case

Our original goal was to produce a complete assurance case for a GIP. Even with help from our colleagues at Medtronic, the University of Pennsylvania, and the Food and Drug Administration, this proved to be a daunting task. In the end, we decided to limit our case to a key aspect of the GIP—its programming by the caregiver.

The case is quite complex and we can’t begin to do it justice in the space allotted. So, rather than attempt to do so, we’ll only present the highlights in this section. The complete assurance case for GIP programming is available in Weinstock [Weinstock 2008]. In this section, we’ll talk about how the assurance case was developed.

There are basically two approaches for structuring a safety assurance case: 1.) focusing on identifying safety requirements and showing that they are satisfied, or 2.) focusing on showing that all safety hazards have been eliminated or adequately mitigated. The approaches are not mutually exclusive —to show that a safety requirement is met one often has to show that hazards defeating the requirement have been eliminated or mitigated. However, each approach has a different flavor. Each has its role to play in developing an assurance case.

Because regulators and manufacturers are used to stating requirements and then ensuring that they are satisfied, top-level claims in an assurance case often have a requirements flavor (e.g., "The GIP is safe") that might be decomposed into sub-claims that the GIP is electrically safe, clinically safe, etc. There is an assumption here that the sub-claims are independent of each other (e.g., that methods for ensuring electrical safety will not interfere with methods for ensuring clinical safety). Unless a case can be decomposed into sub-claims that are relatively independent, the interactions can complicate the case.

Typically, safety requirements arise from an understanding of hazards that need to be addressed; each safety requirement, if satisfied, mitigates one or more hazards. But if the case just addresses safety *requirements*, the link to the hazards mitigated by the requirement can be lost. It can become difficult to decide if the requirement is adequate to address the underlying hazard(s).

For example, infusion pumps are typically battery powered to provide the patient some freedom of movement. An obvious hazard is loss of battery power. So, one might have a safety requirement to help ensure the pump is plugged into electrical power prior to battery exhaustion. Such a requirement might be:

When operating on battery power, visual and auditory alarms are launched at least 10 minutes prior to battery exhaustion but no more than 15 minutes prior.

To demonstrate that this claim holds for a particular infusion pump, we could provide test results showing that warnings are raised at least ten minutes prior to battery exhaustion but no more than fifteen minutes prior. In addition, we could present arguments showing that we have confidence in such test results because the structure of the tests has taken into account various potential causes of misleading results. For example, since the battery discharge profile changes depending on the age of a battery, we would need to show that all the tests were run with a mixture of new and well-used batteries. Similarly, since the electrical load might affect the time to battery exhaustion, we would need to show that the tests were run with different electrical loads on the pump.

Such tests and analyses are fine for demonstrating that the requirement is satisfied, but from a safety viewpoint, we have little documentation about what hazard the requirement is mitigating. In addition, how do we know that ten minutes is the appropriate warning interval for every clinical setting? Is ten minutes enough time for someone to respond the alarm? Will the alarm be heard in every possible setting? How accurate does the measure of remaining power need to be (e.g., is it unacceptable if the alarms are launched when 20 minutes of power remains)? How does this requirement fit with other safety requirements? In short, to fully understand and validate the requirement, we need to establish the larger context within which the requirement exists.

A benefit of focusing on safety requirements is that from a user/regulatory viewpoint, stating the safety requirements and demonstrating that they have been met seems straightforward. However, a safety assurance case that only addresses whether safety requirements are met will focus primarily on what tests and test conditions or other analyses are considered sufficient to show the requirement is met. The case is likely to be less convincing in considering whether all relevant hazards have been met because the reasoning leading from the hazards to the requirement is not necessarily part of the case.

Another problem with a purely requirements-based approach is the difficulty in specifying fault-tolerant behavior. For example, consider a high-level requirement such as: “The GIP delivers the prescribed amount of the prescribed product.” Satisfying this requirement would certainly seem to satisfy a higher level claim that the GIP is safe, but the requirement, as stated, implies that the GIP *always* delivers the right amount of the right product. Clearly, there are factors outside the GIP’s control that can prevent this from happening. For instance, the GIP has no way of recovering when a user enters an unprescribed delivery rate. Similarly, if an infusion line is occluded, the GIP has no way of clearing the line. From a safety viewpoint, we want to ensure the GIP minimizes the chances of harming the patient. Stating a claim that is unachievable in the real world doesn’t allow the case to adequately address safety hazards and their mitigations.

From a safety argument perspective, rather than focusing on safety requirements, it is more convincing to state (and satisfy) hazard mitigation claims (Figure 5-2). For example, a claim such as “The possibility of delivering an incorrect dose has been mitigated” allows the assurance case to discuss the possible hazards resulting from incorrect delivery and then to explain the mitigation approaches, which can include raising alarms to cause a human intervention.

We took the hazards-based approach when developing the GIP assurance case.

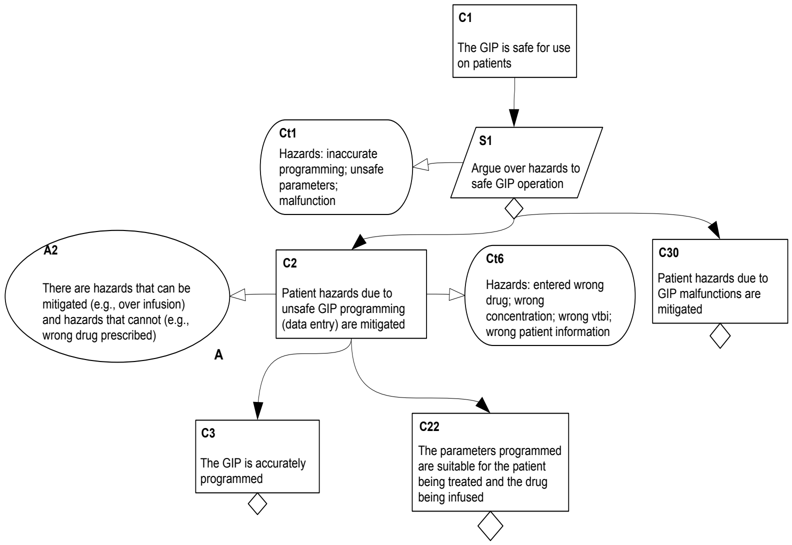

C1
The GIP is safe for use on patients
Ct1
Hazards: inaccurate programming; unsafe parameters; malfunction
S1
Argue over hazards to safe GIP operation
A2
There are hazards that can be mitigated (e.g., over infusion) and hazards that cannot (e.g., wrong drug prescribed)
A
C2
Patient hazards due to unsafe GIP programming (data entry) are mitigated
Ct6
Hazards: entered wrong drug; wrong concentration; wrong vtbi; wrong patient information
C30
Patient hazards due to GIP malfunctions are mitigated
C3
The GIP is accurately programmed
C22
The parameters programmed are suitable for the patient being treated and the drug being infused

Figure 5-2: A Hazard Focused Argument

5.6.3 Medical Device Archetypes and Patterns

Kelly [Kelly 1998] defines the concept of a safety case pattern as a template (with usage instructions) that captures acceptable ways of structuring generic safety arguments. For medical device assurance cases, it would be helpful if a set of agreed argumentation patterns, or archetypes, were available for use by medical device manufacturers and reviewers. If such patterns were provided for different aspects of a particular device's assurance case, they would help to show manufacturers and reviewers how to make effective use of assurance case technology. Since the patterns would provide examples, a barrier to adopting goal-structuring notation would be reduced.

An example of an assurance case archetype arguing that entry errors caused by keypad design are mitigated is presented in Figure 5-3.

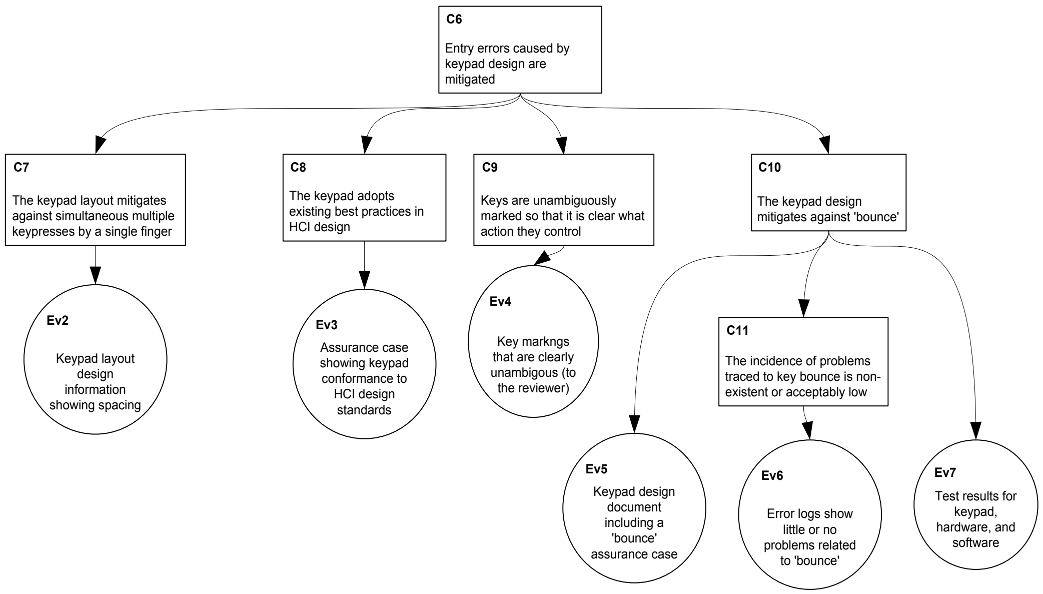

C6
Entry errors caused by keypad design are mitigated
C7
The keypad layout mitigates against simultaneous multiple keypresses by a single finger
C8
The keypad adopts existing best practices in HCI design
C9
Keys are unambiguously marked so that it is clear what action they control
C10
The keypad design mitigates against 'bounce'
Ev2
Keypad layout design information showing spacing
Ev3
Assurance case showing keypad conformance to HCI design standards
Ev4
Key markngs that are clearly unambigous (to the reviewer)
C11
The incidence of problems traced to key bounce is non-existent or acceptably low
Ev5
Keypad design document including a 'bounce' assurance case
Ev6
Error logs show little or no problems related to 'bounce'
Ev7
Test results for keypad, hardware, and software

Figure 5-3: Archetype for Keyboard Entry Errors

From the FDA's perspective, an assurance case containing a pattern or archetype that has already been used in an approved design allows it to certify based on the evidence submitted without necessarily examining the whole argument. This provides more time-savings and more confidence in the approval. For the keyboard design, the list of evidence would include the HCI assurance case, test results, documentation that the "bounce" problem had been addressed, etc. This seems very similar to the process-based checklist, except that each item of evidence fills a specific need in the assurance case pattern that it supports. Further, the FDA could, if necessary, refer to that pattern to see how the evidence specifically supports the claim. But the fact is, given that the pattern has been used successfully before, all the examiner need do is ensure that the assumptions under which the case still hold. If so, he only need consider the individual pieces of evidence.

5.6.4 Concluding Thoughts

Adopting assurance cases into the FDA certification process is going to take work by interested parties. There are forward-thinking manufacturers, FDA personnel, and people at the industry advocacy organization, AdvaMed, who appear ready and willing to make this happen. There are two activities that should be undertaken to ease the transition of assurance cases into the medical device community. The first is to increase industry awareness as to what assurance cases are, and what benefits might derive from their use. A viable approach to this would be to write a series of articles for industry trade magazines that explore the assurance case approach and what its adoption could mean to the industry. Artifacts used in the articles should be industry-related. To be effective, the articles would have to be co-authored by people in the industry. The second activity would be to create and publish a series of FDA-approvable archetypes for different kinds of medical devices.

5.7 References/Bibliography

[Arney 2008]
Arney, David; Jetley, Raoul; Jones, Paul; Lee, Insup & Sokolsky, Oleg. *Generic Infusion Pump Hazard Analysis and Safety Requirements* (MS-CIS-08-31). Philadelphia, PA: University of Pennsylvania, 2008.

[Goodenough 2008]
Goodenough, John B. & Weinstock, Charles B. *Hazards to Evidence: Demonstrating the Quality of Evidence in an Assurance Case* (CMU/SEI-2008-TN-016, in preparation). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2008.

[Kelly 1998]
Kelly, Timothy Patrick. “Arguing Safety – A Systematic Approach to Safety Case Management.” PhD diss., University of York, Department of Computer Science, 1998.

[Weinstock 2008]
Weinstock, Charles B. & Goodenough, John B. Assurance Cases for Medical Devices (CMU/SEI-2008-TN, in preparation). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2008.

6 Modeling Stakeholder Requirements for Integrated Use in Both Process Improvement and Product Development

Robert Stoddard and Robert Nord

6.1 Purpose

This report summarizes the collaboration between the Software Engineering Institute (SEI) Software Engineering Measurement and Analysis (SEMA) team and the Software Architecture Technology (SAT) team on a topic that transcends both the process improvement and product development domains, namely, the modeling of stakeholder requirements for both process improvement and product development. Members from both teams participated in this IRAD project, further defining stakeholder modeling to support both domains and to enable the synergistic use of the varied stakeholders voices in each domain. As will be seen in more detail in the next section, this IRAD was conceived by one of the authors who experienced disjointed modeling of stakeholder requirements, specifically non-alignment of process improvement activities, that resulted in non-optimal product solutions caused.

6.2 Background

During the past fifteen years, the SEI Software Architecture Technology (SAT) team created an invaluable resource of software architecture and product line knowledge codified in a number of books and associated training courses. A non-exhaustive list of topics includes

- Software Architecture Principles [Bass 2003]
- Software Architecture Documentation [Clements 2003]
- Quality Attribute Workshop (QAW) [Barbacci 2003]
- Architecture Attribute-Driven Design (ADD) Method [Wojcik 2006]
- Cost Benefit Analysis Method (CBAM) [Bass 2003]
- Architecture Tradeoff Analysis Method (ATAM) [Clements 2002a]
- Architecture Improvement Workshop (AIW) [Kazman 2006]
- Software Product Lines [Clements 2002b]

From initial application in industry, the authors found these topics to be quite timely in addressing business and technical needs and convincing executives of the need for more discipline and training in systems engineering, software architecture, and product line planning. For a number of industry clients, these topics represented structure where no structure existed before. For many, these topics crystallized what was earlier deemed the “fuzzy front end” of product development. It is from this perspective and experience that the authors identified opportunities to integrate the SEI advancements with existing industry product development lifecycle frameworks [Kazman 2003]. This integration approach extended to include Six Sigma, with the advent of industry adopting the product development methodology called “Design for Six Sigma.” A short synopsis of the evolution of Six Sigma follows to set the stage for this research project.

During approximately the past twenty years, Six Sigma evolved from a pure quality initiative within manufacturing to a complete business governance model including process, tool, and statistical analysis enhancements across the business. Traditional activities involving product portfolio, in-bound technical marketing, research and development, product engineering, supply chain, and out-bound sales and marketing began to use the Six Sigma toolkit and reap the optimization benefits. A key aspect of this evolution is that Six Sigma began with a process improvement perspective originally outlined by Motorola as the 12 steps to Six Sigma, which were then refined and popularized by General Electric as the 5 steps of DMAIC (Define, Measure, Analyze, Improve, and Control). Generally speaking, the vast number of industry articles published in the literature depicted the use of DMAIC in process improvement scenarios. However, a major shift occurred approximately eight years ago, when industry first picked up on the concept of applying the Six Sigma techniques to product development in a formal methodology called "Design for Six Sigma." At this point, product development became the central theme and focus. The shift was dramatic in several ways:

1. Six Sigma methods became more clearly linked and aligned to customer needs and business goals via the products involved.
2. Six Sigma methods and expertise moved out of the almost exclusive club of quality and process improvement professionals to the marketing, product engineering, supply chain, product test and sales professionals.
3. Product scorecards using Six Sigma measurements became the basis for governing the business.
4. Process improvement became more subsidiary and aligned to product development, thus dramatically changing the management sponsorship and targeting of process improvement.

It was during an experience of this shift at Motorola in 2003-04 that one of the authors noted a fundamental change and an opportunity to better integrate process improvement and product development via the modeling of stakeholder requirements. This concept was sparked after the author attended the SEI Software Architecture and Product Line classes in the spring of 2004. For example, it clearly made no sense to continue developing product, architecture and technology roadmaps, at best loosely coupled, and at worst, in isolation. It also made no sense to continue developing process improvement roadmaps that were not closely linked to product and technology roadmaps. Likewise for the planning related to the human resources, including skills development and training, and the development environments and tools.

6.3 Approach

The approach used in this study relied on a combination of literature search, cross-collaboration of experts from several SEI technology groups within a series of creative exploration sessions, and face-to-face workshops with an industry expert from Motorola. A detailed work flow process diagram served to structure the discussions and help identify synergies among several of the technologies (software architecture, software product lines, and Design for Six Sigma). Once the creative sessions were completed, a workshop with the industry expert followed. The workshop aimed to confirm the creative synergies identified by the SEI and to gain greater insight to the early in-bound marketing analysis of the different stakeholders' requirements.

6.4 Collaborations

The SEI IRAD participants were Robert Stoddard and Robert Nord. Additionally, Paul Clements, Mark Klein, and Len Bass participated in the series of work sessions to share insight and create and confirm ideas of synergy between the technologies. The industry expert who collaborated in the workshop was Scott McGregor of McGregor Excellence Consulting, formerly a senior director of Six Sigma for Sales and Marketing at Motorola.

6.5 Evaluation Criteria

The success criteria for judging the results of this IRAD were

- shared common understanding of key aspects of software architecture, software product lines, and Design for Six Sigma among the SEI participants and the industry expert
  - We held a series of orientation, brainstorming, and idea creation sessions that included enlightening discussions on how several of the technologies evolved in similar and different ways, which led to some of the noted strengths of each of the technologies.
- detailed a process flow model to highlight the synergies among the technologies
  - Excerpts of the model are included in this report and the complete model is referenced in the Software Engineering Institute SEMA external website [Stoddard 2008].
- Recommended process enhancements and changes within each technology that would enable the creative synergy of an integrated approach and use of the modeling of stakeholder requirements for both process improvement and product development.

We summarize these in our results section. The next step after this IRAD project is to present these concepts at an annual SEI Architecture Technology User Network (SATURN) conference [SATURN 2008] and elicit industry collaborators to pilot the modeling approach outlined in this IRAD.

6.6 Results

The first step involved identifying the high-level relationships among the different stakeholder voices traditionally involved in product development. For this IRAD, the initial perspective came from industry, in which the early start of voice analysis within Design for Six Sigma took root. As shown in Figure 6-1, there are five primary stakeholder voices within the scope of this IRAD. In subsequent work, additional and different stakeholder voices will be included that better reflect the government and defense contracting environment.

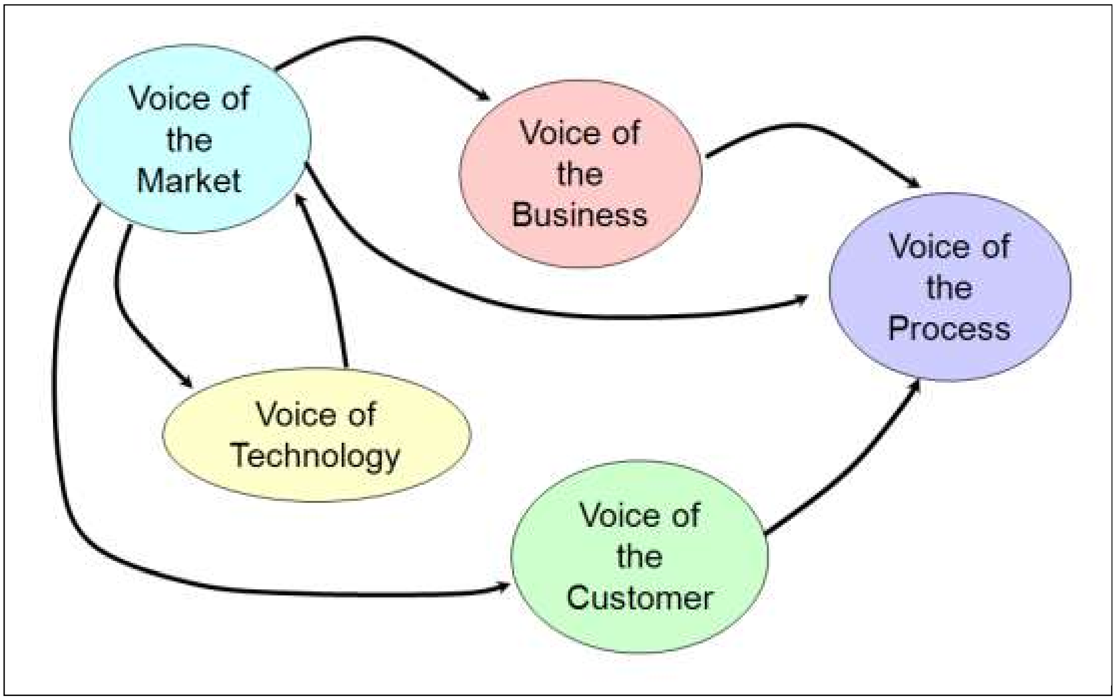


Figure 6-1: The Different Stakeholder Voices

As the different voices were analyzed, we found it useful to discuss the voice activity and analysis in context of process steps outlined in the modern Design for Six Sigma methodology [Creveling 2003, Otto 2001]. The following flow charts represent the leading thoughts on Design for Six Sigma processes. They contain annotations made by the research team about synergies with the other technologies discussed in this IRAD report.

6.6.1 Voice of the Market

The research team first looked at the existing activities described for the Voice of the Market. In addition to the previous references for the flow charts, the team also capitalized on more recent publications detailing the Voice of the Market activity [Creveling 2006]. As may be seen in Figure 6-2, the team annotated that this activity would create outputs that could serve as inputs to the Team Software Process Launch. This could represent an opportunity to further define these inputs to the Team Software Process in a way that capitalized on the Six Sigma analysis during the team launch [Humphrey 2000.] Additionally, a number of works in the past five years defined a number of measures associated with the Voice of the Market activity that represent not only outputs of this Voice but inputs to the other Voices [Westarp 2003, Farris 2006, Davis 2007]. As may be seen in the primary Voice of the Market reference [Creveling 2006], a significant portfolio and in-bound marketing analysis is performed to fully characterize the various market segments, products and associated investments, revenues, and profits.

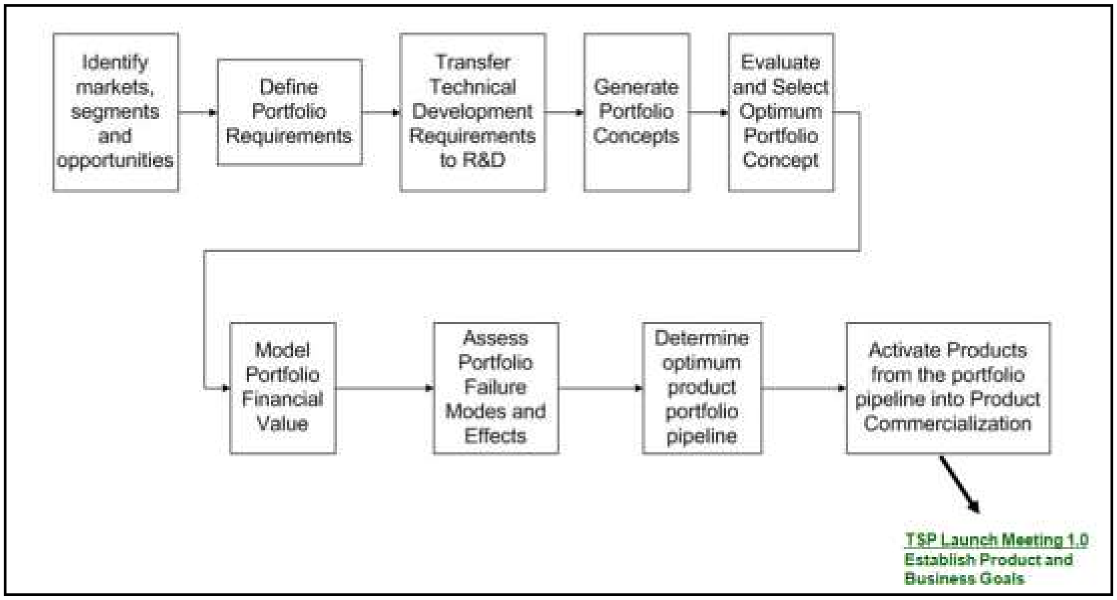

Identify markets, segments and opportunities
Define Portfolio Requirements
Transfer Technical Development Requirements to R&D
Generate Portfolio Concepts
Evaluate and Select Optimum Portfolio Concept
Model Portfolio Financial Value
Assess Portfolio Failure Modes and Effects
Determine optimum product portfolio pipeline
Activate Products from the portfolio pipeline into Product Commercialization
TSP Launch Meeting 1.0
Establish Product and Business Goals

Figure 6-2: The Voice of the Market Process

6.6.2 Voice of the Business

The team noted that extensive work is published on the Voice of the Business. Most notably, the work surrounding the Balanced Scorecard concept, and associated measures and analysis, represents a very mature body of knowledge [Kaplan 1996]. The work most recently was extended to include the concept of strategy maps for executive strategic thinking [Kaplan 2004]. Due the mature body of knowledge and literature on this particular Voice, the team decided to concentrate on the other Voices and then discuss a unifying framework and modeling approach.

6.6.3 Voice of the Customer

The Voice of the Customer represents an activity that bridges the Voice of the Business, the Voice of the Market, and the Voice of Technology. The three Voices play essential roles in the formulation of the product portfolio and set the foundation for a product-line-specific analysis of customer needs and requirements. For many corporations, this activity bridges the activities of strategic marketing and the marketing analysis specific to a customer base for a particular product or product line. As shown in Figure 6-3, the team identified a number of synergies among the Voice of the Customer activity and specific activities within the software architecture technology [Bass 2003, Clements 2003] and the Team Software Process [Humphrey 2000]. Again, follow-on work will define the specific artifacts and uses associated with the synergy points and represent mutual changes or modifications of the technology process.

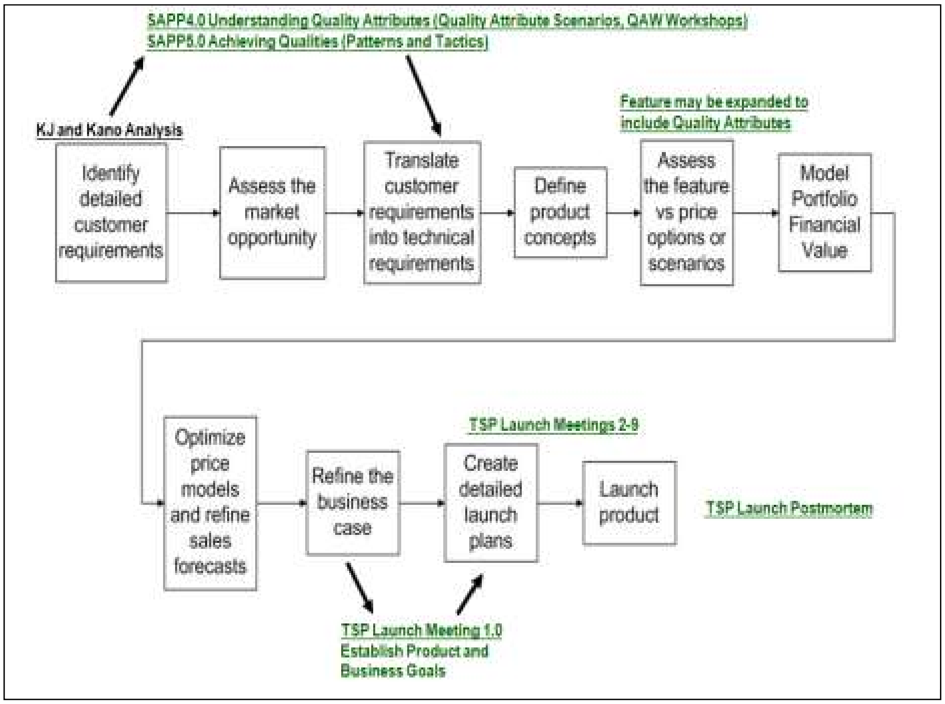

SAPP4.0 Understanding Quality Attributes (Quality Attribute Scenarios, QAW Workshops)
SAPP5.0 Achieving Qualities (Patterns and Tactics)
KJ and Kano Analysis
Feature may be expanded to include Quality Attributes
Identify detailed customer requirements
Assess the market opportunity
Translate customer requirements into technical requirements
Define product concepts
Assess the feature vs price options or scenarios
Model Portfolio Financial Value
TSP Launch Meetings 2-9
Optimize price models and refine sales forecasts
Refine the business case
Create detailed launch plans
Launch product
TSP Launch Postmortem
TSP Launch Meeting 1.0
Establish Product and Business Goals

Figure 6-3: The Voice of the Customer Process

6.6.4 Voice of Technology

The Voice of the Technology activity included much more detailing of process and discussion surrounding synergy points. The following flowcharts are primarily based on standard Design for Six Sigma processes outlined by a leading industry expert [Creveling 2007]. As shown in Figures 6-4 through6-7, the Voice of Technology is discussed in the process known as "Innovate, Ideate, Design, Optimize" (IIDOV). The figures also show a number of synergies with the Software Architecture Technology (SAT) activities [Bass 2003], as well as the Software Product Line Practice (PLP) activities [Clements 2002]. Lastly, Figure 6-7 shows outputs that become inputs to the Team Software Process (TSP) Launch process [Humphrey 2000].

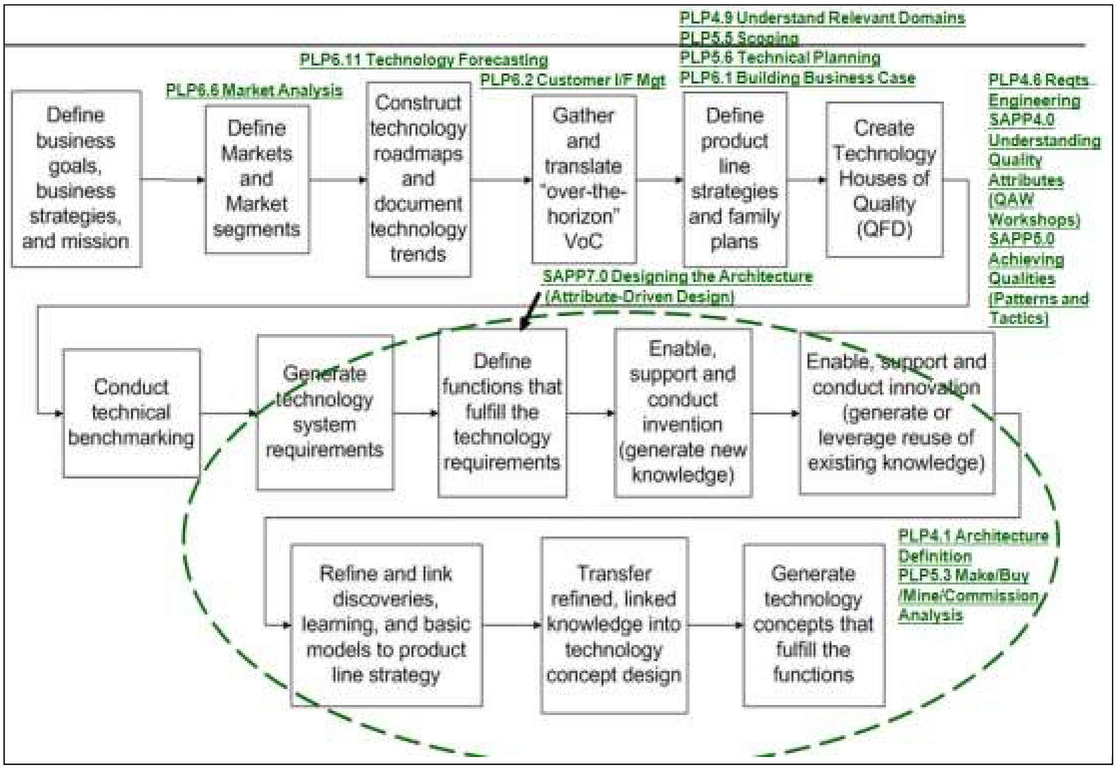

PLP4.9 Understand Relevant Domains
PLP5.5 Scoping
PLP5.6 Technical Planning
PLP6.1 Building Business Case
PLP6.11 Technology Forecasting
PLP6.2 Customer I/F Mgt
PLP6.6 Market Analysis
Define business goals, business strategies, and mission
Define Markets and Market segments
Construct technology roadmaps and document technology trends
Gather and translate "over-the-horizon" VoC
Define product line strategies and family plans
Create Technology Houses of Quality (QFD)
PLP4.6 Reqts. Engineering
SAPP4.0 Understanding Quality Attributes (QAW Workshops)
SAPP5.0 Achieving Qualities (Patterns and Tactics)
SAPP7.0 Designing the Architecture (Attribute-Driven Design)
Conduct technical benchmarking
Generate technology system requirements
Define functions that fulfill the technology requirements
Enable, support and conduct invention (generate new knowledge)
Enable, support and conduct innovation (generate or leverage reuse of existing knowledge)
PLP4.1 Architecture Definition
PLP5.3 Make/Buy /Mine/Commission Analysis
Refine and link discoveries, learning, and basic models to product line strategy
Transfer refined, linked knowledge into technology concept design
Generate technology concepts that fulfill the functions

Figure 6-4: The Voice of Technology Process (Innovate and Ideate)

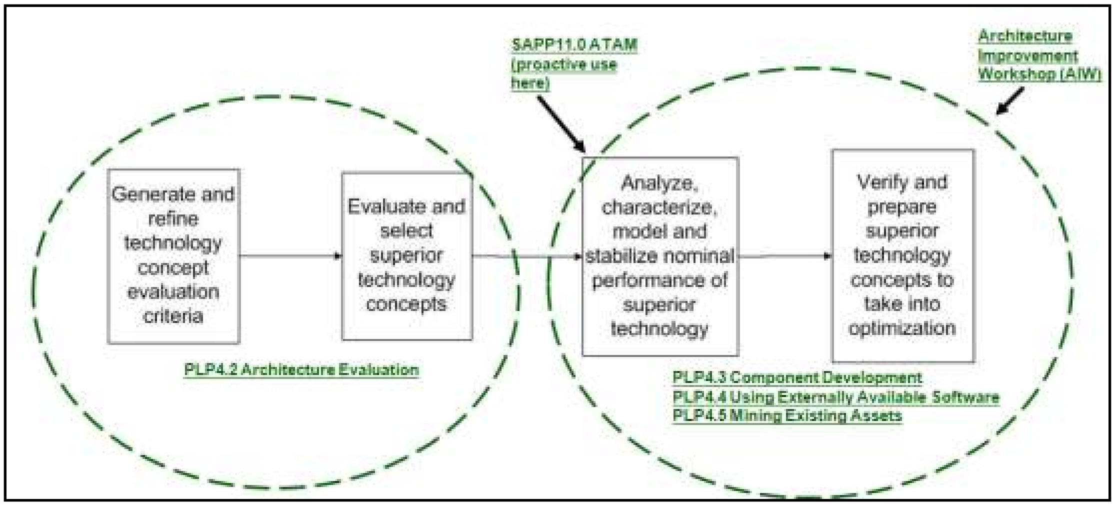

SAPP11.0 ATAM (proactive use here)
Architecture Improvement Workshop (AIW)
Generate and refine technology concept evaluation criteria
Evaluate and select superior technology concepts
Analyze, characterize, model and stabilize nominal performance of superior technology
Verify and prepare superior technology concepts to take into optimization
PLP4.2 Architecture Evaluation
PLP4.3 Component Development
PLP4.4 Using Externally Available Software
PLP4.5 Mining Existing Assets

Figure 6-5: The Voice of Technology Process (Design)

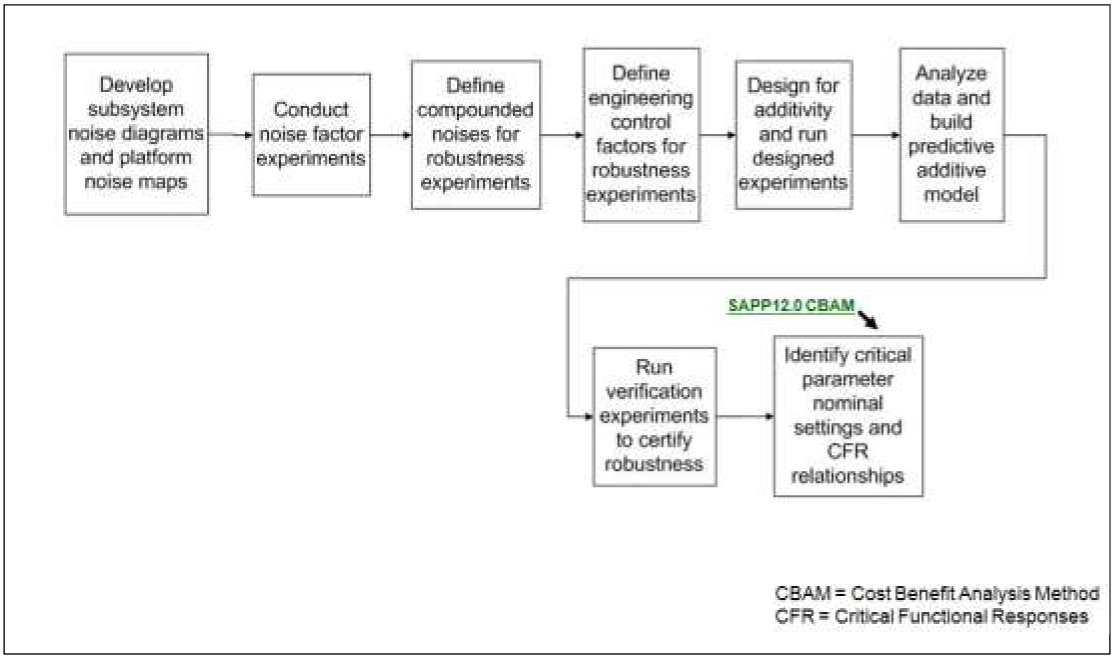

Develop subsystem noise diagrams and platform noise maps
Conduct noise factor experiments
Define compounded noises for robustness experiments
Define engineering control factors for robustness experiments
Design for additivity and run designed experiments
Analyze data and build predictive additive model
SAPP12.0 CBAM
Run verification experiments to certify robustness
Identify critical parameter nominal settings and CFR relationships
CBAM = Cost Benefit Analysis Method
CFR = Critical Functional Responses

Figure 6-6: The Voice of Technology Process (Optimize)

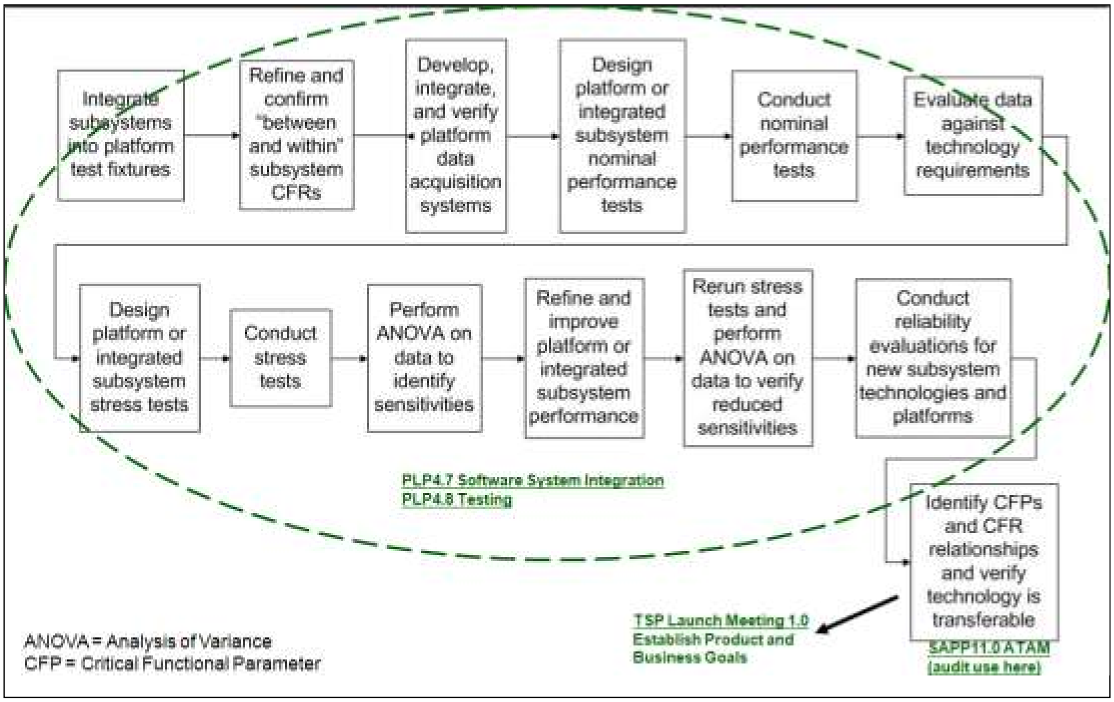

Integrate subsystems into platform test fixtures
Refine and confirm "between and within" subsystem CFRs
Develop, integrate, and verify platform data acquisition systems
Design platform or integrated subsystem nominal performance tests
Conduct nominal performance tests
Evaluate data against technology requirements
Design platform or integrated subsystem stress tests
Conduct stress tests
Perform ANOVA on data to identify sensitivities
Refine and improve platform or integrated subsystem performance
Rerun stress tests and perform ANOVA on data to verify reduced sensitivities
Conduct reliability evaluations for new subsystem technologies and platforms
PLP4.7 Software System Integration
PLP4.8 Testing
Identify CFPs and CFR relationships and verify technology is transferable
TSP Launch Meeting 1.0
Establish Product and Business Goals
SAPP11.0 ATAM
(audit use here)
ANOVA = Analysis of Variance
CFP = Critical Functional Parameter

Figure 6-7: The Voice of Technology Process (Verify)

6.6.5 Voice of the Process

The team also represented the Voice of the Process activity on the leading Design for Sigma process for product development called “Concept, Design, Optimize and Verify” (CDOV) [Creveling 2003]. Again, as may be seen in Figure 6-8 through Figure 6-11, notable synergies were identified with the Software Architecture Technology (SAT) activities [Bass 2003] and the Team Software Process [Humphrey 2000]. In this case, the Team Software Process could be viewed as running somewhat in parallel with many of the activities in this Voice. It is in this Voice activity that the team saw the opportunity to integrate and synergize the activities of traditional Design for Six Sigma, the Software Architecture Technology, and the Team Software Process. The Voice of the Process activity hinges on inputs and knowledge from the other Voices. Essentially, changes or improvements to the process capability must be aligned to support the other Voices. For many organizations, this represents a major change from their traditional approach of driving process improvement projects based on engineering process groups that might be out of touch with the other stakeholder groups’ requirements. With this new approach, both product development and process improvement need to be driven by the commonly understood and integrated set of requirements from all stakeholder groups. This is also congruent with the intent of the SEI CMMI Guidelines for Process Integration and Product Improvement [Chrissis 2007].

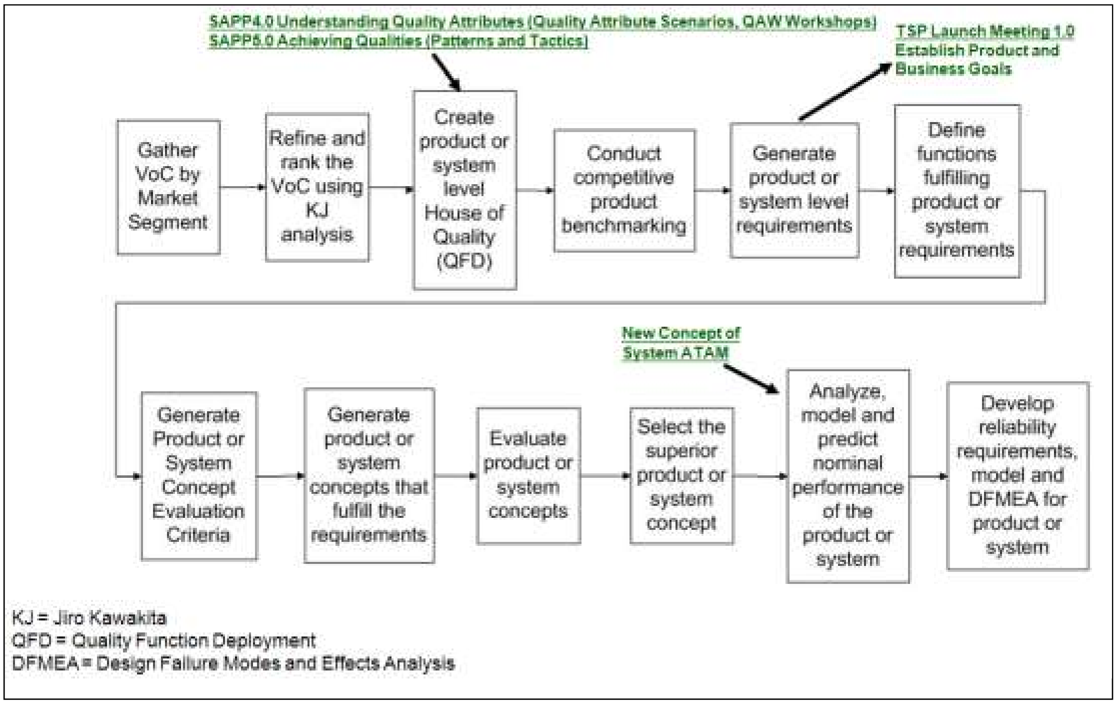

SAPP4.0 Understanding Quality Attributes (Quality Attribute Scenarios, QAW Workshops)
SAPP5.0 Achieving Qualities (Patterns and Tactics)
TSP Launch Meeting 1.0
Establish Product and Business Goals
Gather VoC by Market Segment
Refine and rank the VoC using KJ analysis
Create product or system level House of Quality (QFD)
Conduct competitive product benchmarking
Generate product or system level requirements
Define functions fulfilling product or system requirements
New Concept of System ATAM
Generate Product or System Concept Evaluation Criteria
Generate product or system concepts that fulfill the requirements
Evaluate product or system concepts
Select the superior product or system concept
Analyze, model and predict nominal performance of the product or system
Develop reliability requirements, model and DFMEA for product or system
KJ = Jiro Kawakita
QFD = Quality Function Deployment
DFMEA = Design Failure Modes and Effects Analysis

Figure 6-8: The Voice of Process (Concept)

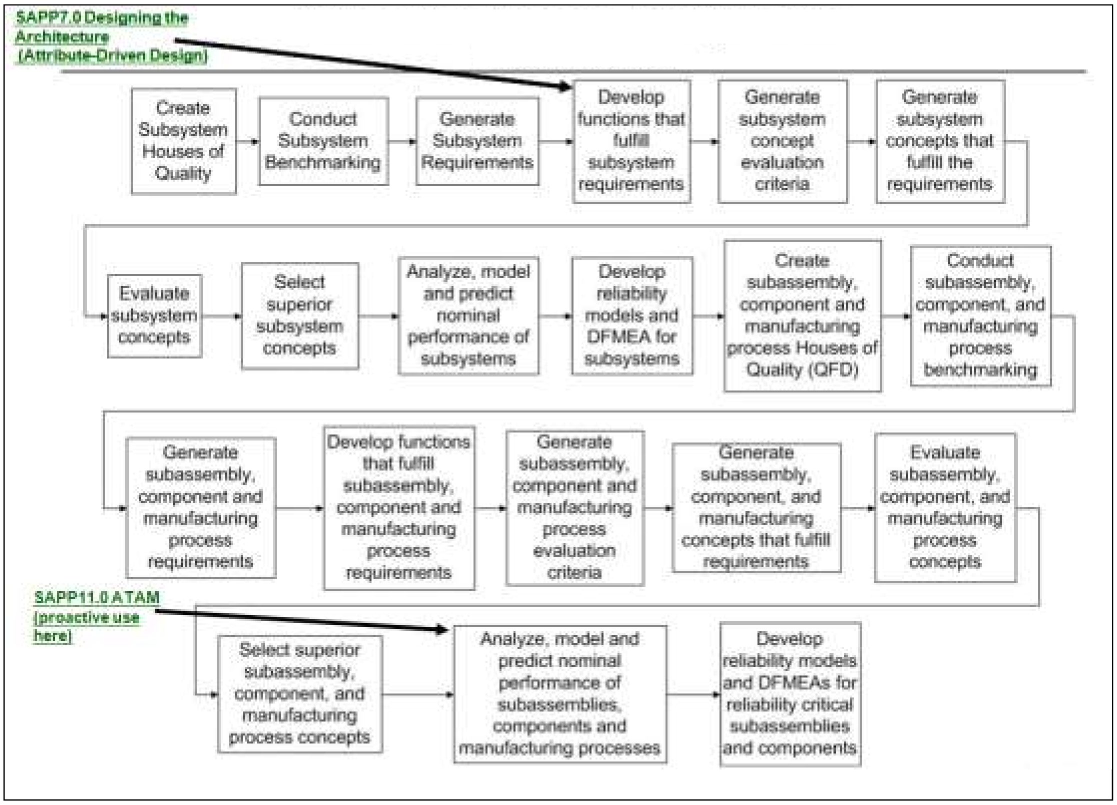

SAPP7.0 Designing the Architecture (Attribute-Driven Design)
Create Subsystem Houses of Quality
Conduct Subsystem Benchmarking
Generate Subsystem Requirements
Develop functions that fulfill subsystem requirements
Generate subsystem concept evaluation criteria
Generate subsystem concepts that fulfill the requirements
Evaluate subsystem concepts
Select superior subsystem concepts
Analyze, model and predict nominal performance of subsystems
Develop reliability models and DFMEA for subsystems
Create subassembly, component and manufacturing process Houses of Quality (QFD)
Conduct subassembly, component, and manufacturing process benchmarking
Generate subassembly, component and manufacturing process requirements
Develop functions that fulfill subassembly, component and manufacturing process requirements
Generate subassembly, component and manufacturing process evaluation criteria
Generate subassembly, component, and manufacturing concepts that fulfill requirements
Evaluate subassembly, component, and manufacturing process concepts
SAPP11.0 ATAM (proactive use here)
Select superior subassembly, component, and manufacturing process concepts
Analyze, model and predict nominal performance of subassemblies, components and manufacturing processes
Develop reliability models and DFMEAs for reliability critical subassemblies and components

Figure 6-9: The Voice of Process (Design)

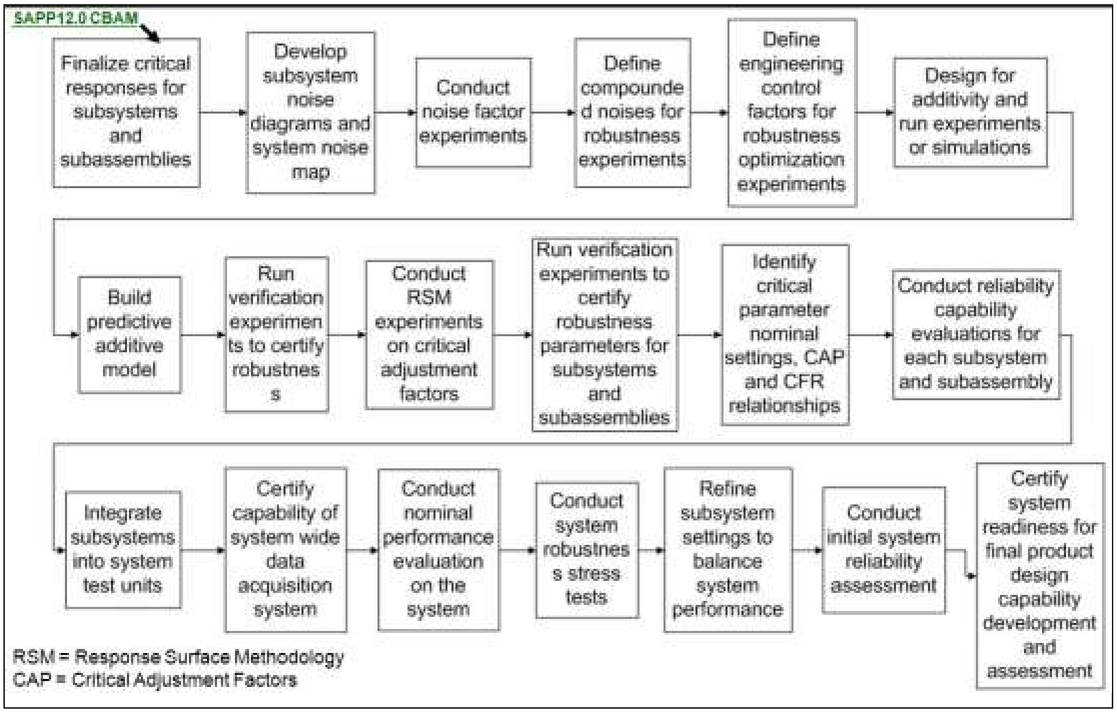

SAPP12.0 CBAM
Finalize critical responses for subsystems and subassemblies
Develop subsystem noise diagrams and system noise map
Conduct noise factor experiments
Define compounded noises for robustness experiments
Define engineering control factors for robustness optimization experiments
Design for additivity and run experiments or simulations
Build predictive additive model
Run verification experiments to certify robustness
Conduct RSM experiments on critical adjustment factors
Run verification experiments to certify robustness parameters for subsystems and subassemblies
Identify critical parameter nominal settings, CAP and CFR relationships
Conduct reliability capability evaluations for each subsystem and subassembly
Integrate subsystems into system test units
Certify capability of system wide data acquisition system
Conduct nominal performance evaluation on the system
Conduct system robustness stress tests
Refine subsystem settings to balance system performance
Conduct initial system reliability assessment
Certify system readiness for final product design capability development and assessment
RSM = Response Surface Methodology
CAP = Critical Adjustment Factors

Figure 6-10: The Voice of Process (Optimize)

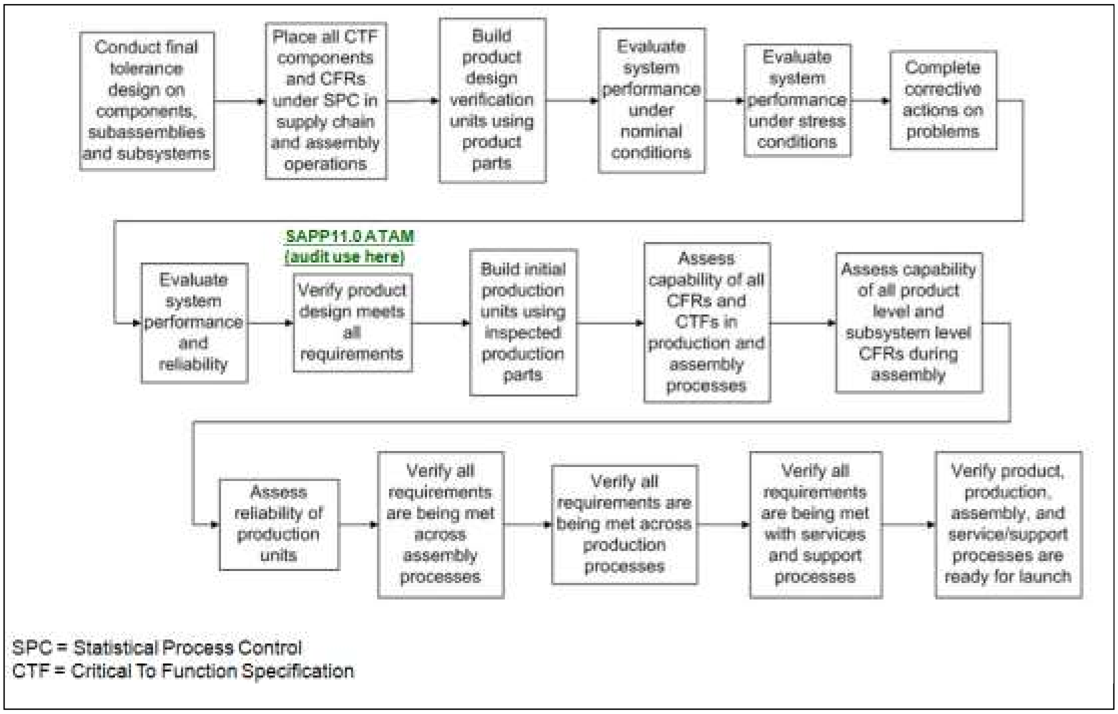

Conduct final tolerance design on components, subassemblies and subsystems
Place all CTF components and CFRs under SPC in supply chain and assembly operations
Build product design verification units using product parts
Evaluate system performance under nominal conditions
Evaluate system performance under stress conditions
Complete corrective actions on problems
Evaluate system performance and reliability
SAPP11.0 ATAM (audit use here)
Verify product design meets all requirements
Build initial production units using inspected production parts
Assess capability of all CFRs and CTFs in production and assembly processes
Assess capability of all product level and subsystem level CFRs during assembly
Assess reliability of production units
Verify all requirements are being met across assembly processes
Verify all requirements are being met across production processes
Verify all requirements are being met with services and support processes
Verify product, production, assembly, and service/support processes are ready for launch
SPC = Statistical Process Control
CTF = Critical To Function Specification

Figure 6-11: The Voice of Process (Verify)

6.6.6 A Unified Voice Framework and Model

Once the team analyzed the modeling of each Voice in detail, its next desire was to promote a framework and modeling approach that would enable a unification of these different Voices. Such a unified approach would have significant benefits:

1. Minimize sub-optimization within a given stakeholder Voice.
2. Enable more informed analysis and modeling within each Voice.
3. Ensure that tradeoffs are properly recognized, evaluated and acted upon.
4. Save extensive iterations and rework that is common in organizations where the different stakeholder Voices are not coordinated and closely coupled.
5. Enable Voices to support each other (e.g. Voice of the Process more aligned with Voice of Technology and Voice of the Customer).

The envisioned unified framework and modeling is actually composed of a hybrid of existing modeling techniques. The team discussed the following modeling techniques that could be joined to enable the desired unified framework and model:

1. Monte Carlo simulation: A practical and industry proven technique to model the uncertainty of factors so that the joint uncertainty of the factors may be seen on one or more outcome measures. Monte Carlo simulation is considered practical as there are several leading

commercial off-the-shelf tools which provide the simulation capability as an add-on capability to Microsoft Excel spreadsheets [Vose 2000].

2. Probabilistic Decision Trees: A modern twist on the age-old concept of decision trees. These trees enable a probabilistic computation on the likelihood of different events including the impact on different decisions. Thus, not only are decisions modeled formally, but a probabilistic likelihood enables a computation on the expected outcomes including expected values of outcomes.
3. Bayesian Belief Networks: Models in which events are associated with each other through either cause and effect relationships or strong correlation [Pourret 2008]. These models are especially beneficial when there is an opportunity to capitalize on both historical information/data as well as current observations. Essentially, these models include conditional probabilities that may be thought of as "the likelihood of an outcome given some other event has occurred." Thus, Bayesian Belief Networks help to model updated predictions as time passes and more information becomes available.
4. Real Options Analysis: Models in which decisions are considered which enable other options or alternative actions to be kept available as long as possible. In other words, this modeling allows analysis of the costs and benefits of taking appropriate actions to keep ones options open in case they are needed. This modeling is most beneficial when uncertainty is significant enough that it could alter decisions about the process, technology, business, product portfolio, etc. Prior to the advent of Real Options analysis, managers would often find themselves locked in to certain outcomes or undesirable events because they unknowingly allowed other opportunities to evaporate. Real Options modeling invokes a more pro-active mindset in anticipating what options or courses of action might become unavailable and then deciding what can be done to keep the alternatives available [Ozkaya 2007].

As may be seen in Figure 6-12, the team envisioned a combined use of the four modeling techniques noted above to address the uncertainty of the analysis within the different Voices and to combine the Voices, model dependencies, and links among the different Voices. At this time, there is not a single toolkit that automates all four of the above modeling techniques, but there are ways to construct an overall modeling framework of the different Voices in which the Voice models may be linked. There are now applications of creating probabilistic decision trees within a Microsoft Excel spreadsheet and then performing Monte Carlo simulation on top of the trees. Other tools, such as Treeage (TreeAge Software, Inc.) approximate some of this capability. And, there remains the ability for wrapper code to be written which would cause communication and data passing between one tool environment and another.

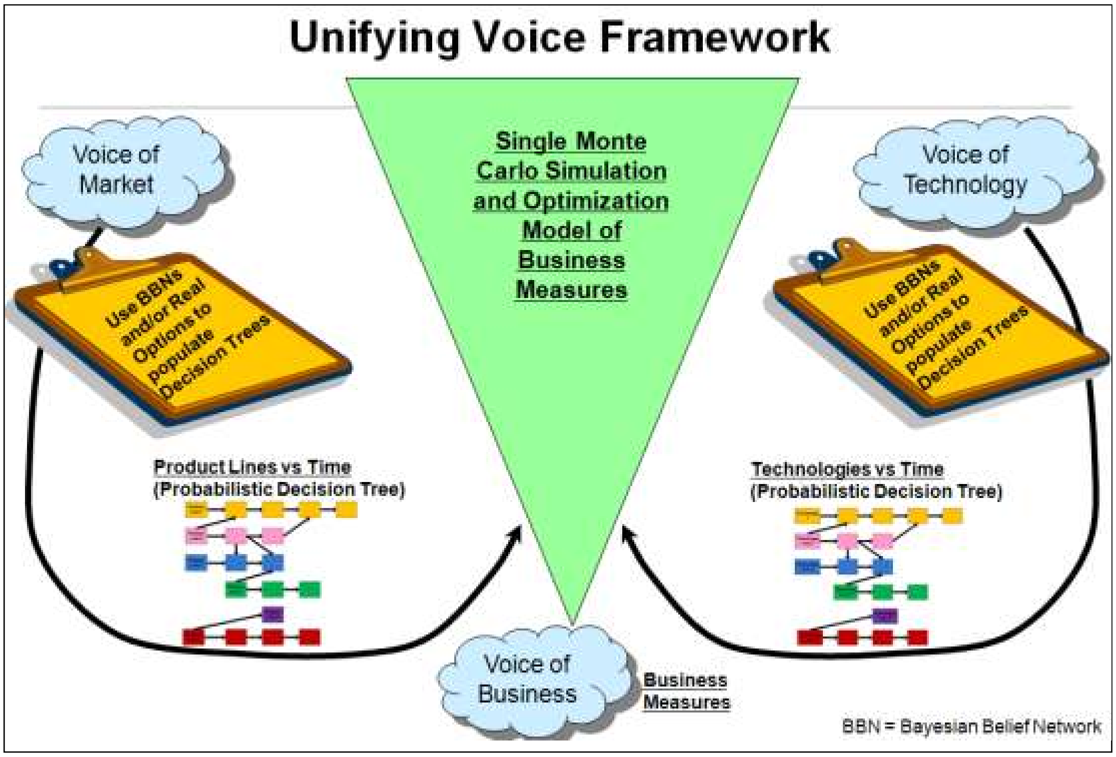

Unifying Voice Framework
Voice of Market
Use BBNs and/or Real Options to populate Decision Trees
Single Monte Carlo Simulation and Optimization Model of Business Measures
Voice of Technology
Use BBNs and/or Real Options to populate Decision Trees
Product Lines vs Time (Probabilistic Decision Tree)
Technologies vs Time (Probabilistic Decision Tree)
Voice of Business
Business Measures
BBN = Bayesian Belief Network

Figure 6-12: The Unified Voice Framework and Model

It is the desire of the IRAD team members to pursue a pilot implementation of this modeling framework within a client setting. The pilot would offer extended knowledge in areas addressing the following questions:

1. How do we best get the varied tools to work together?
2. What kind of wrapper code could be devised for tool interoperability?
3. What kind of client skills are needed to successfully develop and implement such a modeling framework?
4. How difficult is it to get the different stakeholder Voice activities integrated at the synergy points discussed earlier in this report?
5. What are the real business benefits of this type of optimization of the different stakeholder Voices?
6. How difficult is it to modify some of the activities internal to the given stakeholder Voices so that overall optimization can occur?
7. What are the changes in organizational dynamics and politics by adopting this integrated approach? (Some insight on this may be gleaned from a current work in which it is proposed that project management practices be modified and driven based on the architecture of the product [Paulish 2002]).

6.7 References/Bibliography

URLs are valid as of the publication date of this document.

[Barbacci 2003]

Barbacci, Mario R.; Ellison, Robert J.; Lattanze, Anthony J.; Stafford, Judith A.; Weinstock, Charles B. & Wood, William G. *Quality Attribute Workshops (QAWs), Third Edition* (CMU/SEI-2003-TR-016, ADA418428). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003. http://www.sei.cmu.edu/pub/documents/03.reports/pdf/03tr016.pdf

[Bass 2003]

Bass, Len; Clements, Paul & Kazman, Rick. *Software Architecture in Practice, Second Edition*. Boston, MA: Addison-Wesley Publishers, 2003 (ISBN: 0321154959).

[Chrissis 2007]

Chrissis, Mary Beth; Konrad, Mike & Shrum, Sandy. *CMMI: Guidelines for Process Integration and Product Improvement, 2nd edition*. Upper Saddle River, NJ: Addison-Wesley Publishers, 2007 (ISBN: 0321279670).

[Clements 2002a]

Clements, Paul; Kazman, Rick & Klein, Mark. *Evaluating Software Architectures: Methods and Case Studies*. Boston, MA: Addison-Wesley Publishers, 2002 (ISBN: 020170482X).

[Clements 2002b]

Clements, Paul & Northrop, Linda. *Software Product Lines: Practices and Patterns*. Boston, MA: Addison-Wesley Publishers, 2002 (ISBN: 0201703327).

[Clements 2003]

Clements, Paul. *Documenting Software Architectures: Views and Beyond*. Boston, MA: Addison-Wesley Publishers, 2003 (ISBN: 0201703726).

[Creveling 2003]

Creveling, Clyde M.; Slutsky, Jeff & Antis, D. *Design for Six Sigma in Technology and Product Development*. Upper Saddle River, N.J.: Prentice Hall, 2003 (ISBN: 0130092231). http://www.loc.gov/catdir/toc/fy0713/2002027435.html

[Creveling 2006]

Creveling, Clyde M., Hambleton, Lynne & McCarthy, Burke. *Six Sigma for Marketing Processes: An Overview for Marketing Executives, Leaders, and Managers*. Upper Saddle River, NJ: Prentice Hall, 2006 (ISBN: 013199008X). http://www.loc.gov/catdir/toc/ecip062/2005030766.html

[Creveling 2007]

Creveling, Clyde M.. *Six Sigma for Technical Processes: An Overview for R&D Executives, Technical Leaders, and Engineering Managers*. Upper Saddle River, NJ: Prentice Hall, 2007 (ISBN: 0132382326). http://www.loc.gov/catdir/toc/ecip0619/2006027554.html

[Davis 2007]
Davis, John. Measuring Marketing: 103 Key Metrics Every Marketer Needs. Singapore ; Hoboken, NJ: John Wiley & Sons (Asia), 2007 (ISBN: 0470821329).
http://www.loc.gov/catdir/enhancements/fy0714/2007295268-d.html

[Farris 2006]
Farris, Paul. *Marketing Metric : 50+ Metrics Every Executive Should Master*. Upper Saddle River, N.J.: Wharton School Pub., 2006 (ISBN: 0131873709).
http://www.loc.gov/catdir/toc/ecip062/2005031114.html

[Humphrey 2000]
Humphrey, Watts S. *Introduction to the Team Software Process*. Reading, MA: Addison-Wesley Publishers, 2000 (ISBN: 020147719X).

[Kaplan 1996]
Kaplan, R. & Norton, D. *The Balanced Scorecard: Translating Strategy into Action*. Cambridge, MA: Harvard Business School Press, 1996.

[Kaplan 2004]
Kaplan, R. & Norton, D. *Strategy Maps*. Cambridge, MA: Harvard Business School Press, 2004.

[Kazman 2003]
Kazman, Rick; Nord, Robert & Klein, Mark. *A Life-Cycle View of Architecture Analysis and Design Methods* (CMU/SEI-2003-TN-026, ADA421679). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2003.
http://www.sei.cmu.edu/pub/documents/03.reports/pdf/03tn026.pdf

[Kazman 2006]
Kazman, Rick; Bass, Len & Klein, Mark. “The Essential Components of Software Architecture Design and Analysis.” *Journal of Systems and Software 79* 8 (August 2006): 1207-1216.

[Mun 2002]
Mun, Johnathan. *Real options analysis : tools and techniques for valuing strategic investments and decisions*. New York: John Wiley & Sons, 2002 (ISBN: 047125696X).
http://www.loc.gov/catdir/bios/wiley045/2002008978.html

[Otto 2001]
Otto, Kevin N. & Wood, Kristin L. *Product Design: Techniques in Reverse Engineering and New Product Development*. Upper Saddle River, NJ: Prentice Hall, 2001 (ISBN: 0130212717).

[Ozkaya 2007]
Ozkaya, Ipek; Kazman, Rick & Klein, Mark. *Quality-Attribute-Based Economic Valuation of Architectural Patterns* (CMU/SEI-2007-TR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.
http://www.sei.cmu.edu/pub/documents/07.reports/07tr003.pdf

[Paulish 2002]
Paulish, D. J. *Architecture-Centric Software Project Management: A Practical Guide*. Boston, MA: Addison-Wesley Publishers, 2002 (ISBN: 0201734095).

[Pourret 2008]
Pourret, Olivier; Naïm, Patrick & Marcot, Bruce. *Bayesian Networks: A Practical Guide to Applications*. Chichester, West Sussex, Eng.; Hoboken, NJ: John Wiley, 2008 (ISBN: 9780470060308). http://www.loc.gov/catdir/enhancements/fy0804/2007045556-d.html

[SATURN 2008]
SEI Architecture Technology User Network (SATURN), 2008.
http://www.sei.cmu.edu/architecture/saturn/

[Stoddard 2008]
Stoddard, R.; "Visio Model for Modeling Stakeholder Requirements," 2008.
http://www.sei.cmu.edu/sema/presentations.html

[TreeAge Software Inc]
Decision Analysis Software, 2008. http://www.treeage.com/

[Vose 2000]
Vose, David. *Quantitative risk analysis: a guide to Monte Carlo simulation Modeling*. Chichester; New York, NY: Wiley, 2000 (ISBN: 0471958034).
http://www.loc.gov/catdir/enhancements/fy0607/96025714-b.html

[Westarp 2003]
Westarp, Falk von. *Modeling software markets: empirical analysis, network simulations, and marketing implications*. Heidelberg; New York: Physica-Verlag, 2003 (ISBN: 3790800090).
http://www.loc.gov/catdir/enhancements/fy0817/2002042868-d.html

[Wojcik 2006]
Wojcik, Rob; Bachmann, Felix; Bass, Len; Clements, Paul C.; Merson, Paulo; Nord, Robert L. & Wood, William G.. *Attribute-Driven Design (ADD), Version 2.0* (CMU/SEI-2006-TR-023, ADA460414). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.
http://www.sei.cmu.edu/publications/documents/06.reports/06tr023.html

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. **AGENCY USE ONLY** (Leave Blank)	2. **REPORT DATE** December 2008	3. **REPORT TYPE AND DATES COVERED** Final

4. **TITLE AND SUBTITLE**

Results of SEI Independent Research and Development Projects

5. **FUNDING NUMBERS**

FA8721-05-C-0003

6. **AUTHOR(S)**

Dio de Niz, Sherman Eagles, Peter H. Feiler, John Goodenough, Jörgen Hansson, Paul Jones, Rick Kazman, Mark Klein, Prof Insup Lee, Gabriel Moreno, Robert Nord, Ipek Ozkaya, Daniel Plakosh, Raj Rajkumar, Lui Sha, Robert Stoddard, Kurt Wallnau, Charles B. Weinstock, and Lutz Wrage

7. **PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

8. **PERFORMING ORGANIZATION REPORT NUMBER**

CMU/SEI-2008-TR-025

9. **SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

HQ ESC/XPK
5 Eglin Street
Hanscom AFB, MA 01731-2116

10. **SPONSORING/MONITORING AGENCY REPORT NUMBER**

ESC-TR-2008-025

11. **SUPPLEMENTARY NOTES**

12A **DISTRIBUTION/AVAILABILITY STATEMENT**

Unclassified/Unlimited, DTIC, NTIS

12B **DISTRIBUTION CODE**

13. **ABSTRACT (MAXIMUM 200 WORDS)**

The Software Engineering Institute (SEI) annually undertakes several independent research and development (IRAD) projects. These projects serve to (1) support feasibility studies investigating whether further work by the SEI would be of potential benefit and (2) support further exploratory work to determine whether there is sufficient value in eventually funding the feasibility study work as an SEI initiative. Projects are chosen based on their potential to mature and/or transition software engineering practices, develop information that will help in deciding whether further work is worth funding, and set new directions for SEI work. This report describes the IRAD projects that were conducted during fiscal year 2008 (October 2007 through September 2008).

14. **SUBJECT TERMS**

independent research and development, IRAD, architectural design decisions, software architecture, software-intensive systems, real-time systems, embedded systems, fault tolerance, fault management, fault containment, Vickrey-Clarke-Groves, auction mechanism, VCG, computational mechanism design

15. **NUMBER OF PAGES**

66

16. **PRICE CODE**

17. **SECURITY CLASSIFICATION OF REPORT** Unclassified	18. **SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	19. **SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	20. **LIMITATION OF ABSTRACT** UL